REJ10J1130-0900

Everywhere you imagine. RENESAS

H8SX E6000H Emulator

User's Manual

Renesas Microcomputer Development Environment System
H8SX Family / H8SX/1650 Group
H8SX/1520 Group
H8SX/1544 Group
HS1650EPH60HE

User's Manual

Rev.9.00
Revision Date: Jun. 13, 2006

Renesas Technology
www.renesas.com

Keep safety first in your circuit designs!

1. Renesas Technology Corp. puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage.
   Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corp. product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corp. or a third party.
2. Renesas Technology Corp. assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corp. without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor for the latest product information before purchasing a product listed herein.
   The information described here may contain technical inaccuracies or typographical errors.
   Renesas Technology Corp. assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
   Please also pay attention to information published by Renesas Technology Corp. by various means, including the Renesas Technology Corp. Semiconductor home page (http://www.renesas.com).
4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corp. assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corp. semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
6. The prior written approval of Renesas Technology Corp. is necessary to reprint or reproduce in whole or in part these materials.
7. If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
   Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
8. Please contact Renesas Technology Corp. for further details on these materials or the products contained therein.

IMPORTANT INFORMATION

READ FIRST

• READ this user's manual before using this emulator product.

• KEEP the user's manual handy for future reference.

Do not attempt to use the emulator product until you fully understand its mechanism.

Emulator Product:

Throughout this document, the term "emulator product" shall be defined as the following products produced only by Renesas Technology Corp. excluding all subsidiary products.

- Emulator station
- PC interface board
- User system interface board
- Cable

The user system or a host computer is not included in this definition.

Purpose of the Emulator Product:

This emulator product is a software and hardware development tool for systems employing the Renesas microcomputer. This emulator product must only be used for the above purpose.

Limited Applications:

This emulator product is not authorized for use in MEDICAL, atomic energy, aeronautical or space technology applications without consent of the appropriate officer of a Renesas sales company. Such use includes, but is not limited to, use in life support systems. Buyers of this emulator product must notify the relevant Renesas sales offices before planning to use the product in such applications.

Improvement Policy:

Renesas Technology Corp. (including its subsidiaries, hereafter collectively referred to as Renesas) pursues a policy of continuing improvement in design, performance, and safety of the emulator product. Renesas reserves the right to change, wholly or partially, the specifications, design, user's manual, and other documentation at any time without notice.

Target User of the Emulator Product:

This emulator product should only be used by those who have carefully read and thoroughly understood the information and restrictions contained in the user's manual. Do not attempt to use the emulator product until you fully understand its mechanism.

It is highly recommended that first-time users be instructed by users that are well versed in the operation of the emulator product.

LIMITED WARRANTY

Renesas warrants its emulator products to be manufactured in accordance with published specifications and free from defects in material and/or workmanship. Renesas, at its option, will repair or replace any emulator products returned intact to the factory, transportation charges prepaid, which Renesas, upon inspection, determine to be defective in material and/or workmanship. The foregoing shall constitute the sole remedy for any breach of Renesas' warranty. See the Renesas warranty booklet for details on the warranty period. This warranty extends only to you, the original Purchaser. It is not transferable to anyone who subsequently purchases the emulator product from you. Renesas is not liable for any claim made by a third party or made by you for a third party.

DISCLAIMER

RENESAS MAKES NO WARRANTIES, EITHER EXPRESS OR IMPLIED, ORAL OR WRITTEN, EXCEPT AS PROVIDED HEREIN, INCLUDING WITHOUT LIMITATION THEREOF, WARRANTIES AS TO MARKETABILITY, MERCHANTABILITY, FITNESS FOR ANY PARTICULAR PURPOSE OR USE, OR AGAINST INFRINGEMENT OF ANY PATENT. IN NO EVENT SHALL RENESAS BE LIABLE FOR ANY DIRECT, INCIDENTAL OR CONSEQUENTIAL DAMAGES OF ANY NATURE, OR LOSSES OR EXPENSES RESULTING FROM ANY DEFECTIVE EMULATOR PRODUCT, THE USE OF ANY EMULATOR PRODUCT, OR ITS DOCUMENTATION, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. EXCEPT AS EXPRESSLY STATED OTHERWISE IN THIS WARRANTY, THIS EMULATOR PRODUCT IS SOLD "AS IS ", AND YOU MUST ASSUME ALL RISK FOR THE USE AND RESULTS OBTAINED FROM THE EMULATOR PRODUCT.

State Law:

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may have other rights which may vary from state to state.

The Warranty is Void in the Following Cases:

Renesas shall have no liability or legal responsibility for any problems caused by misuse, abuse, misapplication, neglect, improper handling, installation, repair or modifications of the emulator product without Renesas' prior written consent or any problems caused by the user system.

All Rights Reserved:

This user's manual and emulator product are copyrighted and all rights are reserved by Renesas. No part of this user's manual, all or part, may be reproduced or duplicated in any form, in hard-copy or machine-readable form, by any means available without Renesas' prior written consent.

Other Important Things to Keep in Mind:

1. Circuitry and other examples described herein are meant merely to indicate the characteristics and performance of Renesas' semiconductor products. Renesas assumes no responsibility for any intellectual property claims or other problems that may result from applications based on the examples described herein.

2. No license is granted by implication or otherwise under any patents or other rights of any third party or Renesas.

Figures:

Some figures in this user's manual may show items different from your actual system.

Limited Anticipation of Danger:

Renesas cannot anticipate every possible circumstance that might involve a potential hazard. The warnings in this user's manual and on the emulator product are therefore not all inclusive. Therefore, you must use the emulator product safely at your own risk.

SAFETY PAGE

READ FIRST

• READ this user's manual before using this emulator product.

• KEEP the user's manual handy for future reference.

Do not attempt to use the emulator product until you fully understand its mechanism.

DEFINITION OF SIGNAL WORDS

This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

⚠ DANGER **DANGER** indicates an imminently hazardous situation which, if not avoided, will result in death or serious injury.

⚠ WARNING **WARNING** indicates a potentially hazardous situation which, if not avoided, could result in death or serious injury.

⚠ CAUTION **CAUTION** indicates a potentially hazardous situation which, if not avoided, may result in minor or moderate injury.

CAUTION **CAUTION** used without the safety alert symbol indicates a potentially hazardous situation which, if not avoided, may result in property damage.

NOTE emphasizes essential information.

⚠ WARNING

Observe the precautions listed below. Failure to do so will result in a FIRE HAZARD and will damage the user system and the emulator product or will result in PERSONAL INJURY. The USER PROGRAM will be LOST.

1. **Carefully handle the emulator product to prevent receiving an electric shock because the emulator product has a DC power supply. Do not repair or remodel the emulator product by yourself for electric shock prevention and quality assurance.**
2. **Always switch OFF the emulator and user system before connecting or disconnecting any CABLES or PARTS.**
3. **Always before connecting, make sure that pin 1 on both sides are correctly aligned.**
4. **Supply power according to the power specifications and do not apply an incorrect power voltage. Use only the provided AC power cable. Use only the specified type of fuse.**

Warnings on Emulator Usage

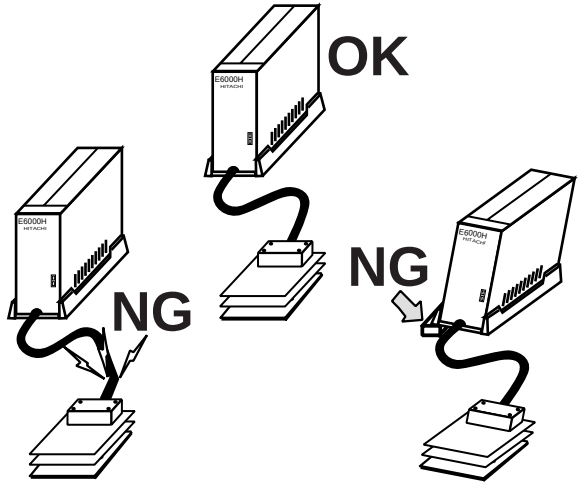
Warnings described below apply as long as you use this emulator. Be sure to read and understand the warnings below before using this emulator. Note that these are the main warnings, not the complete list.

⚠ WARNING

Always switch OFF the emulator and user system before connecting or disconnecting any CABLES or PARTS. Failure to do so will result in a FIRE HAZARD and will damage the user system and the emulator product or will result in PERSONAL INJURY. The USER PROGRAM will be LOST.

CAUTION

Place the emulator station and evaluation chip board so that the trace cables are not bent or twisted. A bent or twisted cable will impose stress on the user interface leading to connection or contact failure. Make sure that the emulator station is placed in a secure position so that it does not move during use nor impose stress on the user interface.



CAUTION

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

Preface

Thank you for purchasing the E6000H emulator.

CAUTION

Read this manual before using the emulator product. Incorrect operation or connection will damage the user system, the emulator product, and the user program.

The E6000H emulator is an efficient software and hardware development support tool for application systems using the microcomputer developed by Renesas Technology Corp.

The E6000H emulator can either be used without a user system, for developing and debugging software, or connected via a user system interface cable to a user system, for debugging user hardware.

The emulator provides the following features:

1. Realtime emulation of the MCU
2. Efficient debugging enabled by variable break functions and a mass-storage trace memory (128-kcycles)
3. Parallel access with a command execution during emulation, for example
   - Trace data display
   - Emulation memory display and modification
4. Performance analysis
   Measurement of subroutine execution time and count for evaluating the execution efficiency of user programs
5. Graphical User Interface by the High-performance Embedded Workshop that runs on Windows® operating systems

High-performance Embedded Workshop is a Graphical User Interface intended to ease the development and debugging of applications written in C/C++ programming language and assembly language. Its aim is to provide a powerful yet intuitive way of accessing, observing and modifying the debugging platform in which the application is running.

The High-performance Embedded Workshop is a powerful development environment for embedded applications targeted at microcontrollers. The main features are:

- A configurable build engine that allows you to set-up compiler, assembler and linker options via an easy to use interface.
- An integrated text editor with user customizable syntax coloring to improve code readability.
- A configurable environment to run your own tools.
- An integrated debugger which allows you to build and debug in the same application.
- Version control support.

The High-performance Embedded Workshop has been designed with two key aims; firstly to provide you, the user, with a set of powerful development tools and, secondly, to unify and present them in a way that is easy to use.

About This Manual

This manual is comprised of two parts: Hardware Part and Debugger Part.

Hardware Part:	Preparation before use, hardware specifications, and troubleshooting procedure.
Debugger Part:	A peculiar debugging function to the emulator, tutorial, emulator software specification, and notes.

This manual describes the debugging function for H8SX E6000H Emulator debugger that used with the High-performance Embedded Workshop.

For detailed information on the basic “look and feel” of the High-performance Embedded Workshop and customizing the High-performance Embedded Workshop environment and the build and the debugging functions common to the High-performance Embedded Workshop products, refer to the High-performance Embedded Workshop user's manual.

This manual does not intend to explain how to write C/C++ or assembly language programs, how to use any particular operating system or how best to tailor code for the individual devices. These issues are left to the respective manuals.

Microsoft, MS-DOS, Windows, Windows NT are registered trademarks of Microsoft Corporation.

Visual SourceSafe is a trademark of Microsoft Corporation.

IBM is a registered trademark of International Business Machines Corporation.

All brand or product names used in this manual are trademarks or registered trademarks of their respective companies or organizations.

Document Conventions

This manual uses the following typographic conventions:

Table 1 Typographic Conventions

Convention	Meaning
[Menu->Menu Option]	Bold text with ‘->’ is used to indicate menu options (for example, **[File->Save As...]**).
FILENAME.C	Uppercase names are used to indicate filenames.
“enter this string”	Used to indicate text that must be entered (excluding the “” quotes).
Key + Key	Used to indicate required key presses. For example, **CTRL+N** means press the **CTRL** key and then, whilst holding the **CTRL** key down, press the **N** key.
➲ (The “how to” symbol)	When this symbol is used, it is always located in the left-hand margin. It indicates that the text to its immediate right is describing “how to” do something.

Components

Check all the components described in the component list unpacking. If the components are not complete, contact a Renesas sales office.

Contents

H8SX/1650 Hardware Part

Section 1 Overview 1
1.1 Notes on Usage 1
1.2 Emulator Hardware Components 2
1.2.1 E6000H Station Components
(A Part of Photos may be Different from Real Appearances) 3
1.2.2 Evaluation Chip Board Configuration 5
1.3 System Configuration 6
1.3.1 System Configuration Using Various Interfaces 6

Section 2 Preparation before Use 9
2.1 Description on Emulator Usage 9
2.2 Emulator Connection 10
2.2.1 Connecting the Emulator to the User System 10
2.2.2 Connecting the User System Interface Board 10
2.2.3 Connection by the User System Interface Connectors 11
2.2.4 Pin Arrangement on the User System Interface Connector 14
2.2.5 Precautions on Connecting the User System 22
2.2.6 Connecting the External Probe (Not Supported in this Emulator) 23
2.2.7 Selecting the Clock 24
2.2.8 Connecting the System Ground 26
2.2.9 PC Interface Board Specifications 27

Section 3 Hardware Specifications 29
3.1 Environmental Conditions 29
3.2 Emulator External Dimensions and Mass 30
3.3 User System Interface Circuit 31
3.3.1 User System Interface Circuit 31
3.4 Support of the Target MCU 39
3.4.1 Memory Space 39
3.4.2 Power-Down Modes (Sleep, Software Standby, and Hardware Standby) 39
3.4.3 Interrupts 39
3.4.4 Control Input Signals (/RES, /NMI, /BREQ, /WAIT, and /STBY) 40
3.4.5 Watchdog Timer (WDT) 40
3.4.6 A/D Converter 40
3.4.7 Emulator State and Internal Modules 41
3.4.8 Differences in Values of Registers 42
3.4.9 Emulation Memory 42
3.5 Notes Specific to the H8SX/1650 E6000H Emulator 43
3.5.1 Custom Device Function 43
3.5.2 Exception Processing of Sleep Instructions 43
3.5.3 Subclock Operation 43

Section 4 Diagnostic Test Procedure 45
4.1 System Set-Up for Diagnostic Program Execution 45
4.2 Test Item of the Diagnostic Program 47
4.3 Diagnostic Test Procedure Using the Diagnostic Program 48

H8SX/1527 and H8SX/1527R Hardware Part

Section 1 Overview 1
1.1 Notes on Usage 1
1.2 Emulator Hardware Components 2
1.2.1 E6000H Station Components
(A Part of Photos may be Different from Real Appearances) 3
1.2.2 Front-end Unit Configuration 5
1.3 System Configuration 6
1.3.1 System Configuration Using Various Interfaces 6

Section 2 Preparation before Use 9
2.1 Description on Emulator Usage 9
2.2 Emulator Connection 10
2.2.1 Connecting the Emulator to the User System 10
2.2.2 Connecting the User System Interface Cable 10
2.2.3 Precautions on Connecting the User System 11
2.2.4 Connecting the External Probe 12
2.2.5 Selecting the Clock 13
2.2.6 Connecting the System Ground 15
2.2.7 PC Interface Board Specifications 16

Section 3 Hardware Specifications 17
3.1 Environmental Conditions 17
3.2 Emulator External Dimensions and Mass 18
3.3 User System Interface Circuit 19
3.3.1 User System Interface Circuit 19
3.4 Support of the Target MCU 25
3.4.1 Memory Space 25
3.4.2 Power-Down Modes (Sleep and Software Standby) 25
3.4.3 Interrupts 25
3.4.4 Control Input Signals (/RES and /NMI) 26
3.4.5 Watchdog Timer (WDT) 26
3.4.6 A/D Converter 26
3.4.7 Emulator State and Internal Modules 27
3.4.8 Differences in Values of Registers 28
3.4.9 Differences in Access to the Internal RAM 28
3.5 Notes Specific to the H8SX/1527 E6000H and H8SX/1527R E6000H Emulators 29
3.5.1 Custom Device Function 29
3.5.2 Non-availability of P2 Open-Drain Outputs 29
3.5.3 Limitations on Control of the SSU Pins 29

Section 4 Diagnostic Test Procedure 31
4.1 System Set-Up for Diagnostic Program Execution 31
4.2 Test Item of the Diagnostic Program (HS1527KEPH60H) 33
4.3 Diagnostic Test Procedure Using the Diagnostic Program 34
4.4 Test Item of the Diagnostic Program (HS1527REPH60H) 37
4.5 Diagnostic Test Procedure Using the Diagnostic Program 38

H8SX/1544 Hardware Part

Section 1 Overview 1
1.1 Notes on Usage 1
1.2 Emulator Hardware Components 2
1.2.1 E6000H Station Components
(A Part of Photos may be Different from Real Appearances) 3
1.2.2 Front-end Unit Configuration 5
1.3 System Configuration 6
1.3.1 System Configuration Using Various Interfaces 6

Section 2 Preparation before Use 9
2.1 Description on Emulator Usage 9
2.2 Emulator Connection 10
2.2.1 Connecting the Emulator to the User System 10
2.2.2 Connecting the User System Interface Cable 10
2.2.3 Precautions on Connecting the User System 11
2.2.4 Connecting the External Probe 12
2.2.5 Selecting the Clock 13
2.2.6 Connecting the System Ground 15
2.2.7 PC Interface Board Specifications 16

Section 3 Hardware Specifications 17
3.1 Environmental Conditions 17
3.2 Emulator External Dimensions and Mass 18
3.3 User System Interface Circuit 19
3.3.1 User System Interface Circuit 19
3.4 Support of the Target MCU 26
3.4.1 Memory Space 26
3.4.2 Power-Down Modes
(Sleep, Software Standby, Hardware Standby, and All-Module-Clock-Stop) 26
3.4.3 Interrupts 26
3.4.4 Control Input Signals (/RES, /NMI, and /STBY) 27
3.4.5 Watchdog Timer (WDT) 27
3.4.6 A/D Converter 27
3.4.7 Emulator State and Internal Modules 28
3.4.8 Differences in Values of Registers 29
3.5 Notes Specific to the H8SX/1544 E6000H Emulator 29
3.5.1 Custom Device Function 29
3.5.2 Subclock Operation 29
3.5.3 Open-Drain Control Registers (PnODR) —Restriction 29
3.5.4 Synchronous Serial Communication Unit (SSU) —Restriction 29
3.5.5 Port Registers (PORTx) —Restriction 30
3.5.6 Input Buffer Control Register (PnICR) —Restriction 30
3.5.7 Watch Timer (WAT) —Restriction 30
3.5.8 Port Function Control Register 4 (PFCR4) —Note 30
3.5.9 Access to the Internal RAM —Note 30
3.5.10 External Expanded Mode —Restrictions 30

Section 4 Diagnostic Test Procedure 31
4.1 System Set-Up for Diagnostic Program Execution 31
4.2 Test Item of the Diagnostic Program 33

4.3 Diagnostic Test Procedure Using the Diagnostic Program 34

Debugger Part

Section 1 Overview 1

Section 2 Preparation before Use 3
2.1 Method for Activating High-performance Embedded Workshop 3
2.1.1 Creating a New Workspace (Toolchain Not Used) 4
2.1.2 Creating a New Workspace (Toolchain Used) 8
2.1.3 Selecting an Existing Workspace 12
2.2 Connecting the Emulator 13
2.3 Re-connecting the Emulator 14
2.4 Ending the Emulator 14

Section 3 Debugging 15
3.1 Setting the Environment for Emulation 15
3.1.1 Opening the [Configuration Properties] Dialog Box 15
3.1.2 Customizing the Settings for the Target MCU 17
3.1.3 Selecting the Interface to be Connected 18
3.1.4 Opening the [Memory Mapping] Dialog Box 19
3.1.5 Changing the Memory Map Setting 20
3.2 Downloading a Program 21
3.2.1 Downloading a Program 21
3.2.2 Viewing the Source Code 21
3.2.3 Viewing the Assembly-Language Code 24
3.2.4 Modifying the Assembly-Language Code 25
3.2.5 Viewing a Specific Address 25
3.2.6 Viewing the Current Program Counter Address 25
3.3 Viewing the Current Status 26
3.4 Reading and Displaying the Emulator Information Regularly 27
3.4.1 Opening the [Extended Monitor] Window 27
3.4.2 Selecting Items to be Displayed 28
3.5 Displaying Memory Contents in Realtime 29
3.5.1 Opening the [Monitor] Window 29
3.5.2 Changing the Monitor Settings 31
3.5.3 Temporarily Stopping Update of the Monitor 31
3.5.4 Deleting the Monitor Settings 31
3.5.5 Monitoring Variables 31
3.5.6 Hiding the [Monitor] Window 32
3.5.7 Managing the [Monitor] Window 33
3.6 Looking at Variables 34
3.6.1 [Watch] Window 34
3.7 Using the Event Points 36
3.7.1 Setting a Software Breakpoint 37
3.7.2 Setting an On-Chip Breakpoint 39
3.7.3 Settings an On-Emulator Breakpoint 42
3.7.4 Editing Event Points 45
3.7.5 Modifying Event Points 45
3.7.6 Enabling an Event Point 45
3.7.7 Disabling an Event Point 45
3.7.8 Deleting an Event Point 46
3.7.9 Deleting All Event Points 46

3.7.10 Viewing the Source Line for an Event Point....46
3.7.11 Setting a Data Condition for a Variable in the Source Program....46
3.8 Viewing the Trace Information....47
3.8.1 Opening the [Trace] Window....47
3.8.2 Acquiring Trace Information....47
3.8.3 Specifying Trace Acquisition Conditions....49
3.8.4 Searching for a Trace Record....56
3.8.5 Clearing the Trace Information....58
3.8.6 Saving the Trace Information in a File....58
3.8.7 Viewing the [Editor] Window....58
3.8.8 Trimming the Source....58
3.8.9 Temporarily Stopping Trace Acquisition....58
3.8.10 Restarting Trace Acquisition....58
3.8.11 Extracting Records from the Acquired Information....59
3.8.12 Calculating the Difference in Time Stamping....62
3.8.13 Analyzing Statistical Information....63
3.8.14 Extracting Function Calls from the Acquired Trace Information....64
3.9 Analyzing Performance....65
3.9.1 Opening the [Performance Analysis] Window....67
3.9.2 Setting Conditions for Measurement....68
3.9.3 Starting Performance Data Acquisition....74
3.9.4 Deleting a Measurement Condition....74
3.9.5 Deleting All Measurement Conditions....74
3.10 Profiling Function....75
3.10.1 Enabling the Profile....75
3.10.2 Specifying Measuring Mode....75
3.10.3 Executing the Program and Checking the Results....75
3.10.4 [List] Sheet....76
3.10.5 [Tree] Sheet....77
3.11 [Profile-Chart] Window....79
3.12 RTOS Extension Function....81
3.12.1 [RTOS Support Function Configuration Properties] Dialog Box....81
3.12.2 Task Step Functions....82
3.12.3 Functions Made Available by [TASK Selection]....83
3.12.4 Performance Measurement (Conditions for Measurement)....84
3.12.5 Event Point (On-Emulator Breakpoint)....84

Section 4 Tutorial....85
4.1 Introduction....85
4.2 Running the High-performance Embedded Workshop....86
4.3 Connecting the Emulator....87
4.3.1 Selecting a Session....87
4.3.2 Connecting the Emulator....87
4.4 Downloading the Tutorial Program....88
4.4.1 Downloading the Tutorial Program....88
4.4.2 Displaying the Source Program....89
4.5 Setting a Software Breakpoint....90
4.6 Setting Registers....91
4.7 Executing the Program....92
4.8 Reviewing Breakpoints....95
4.9 Referring to Symbols....96
4.10 Viewing Memory....97
4.11 Watching Variables....98

4.12 Displaying Local Variables 101
4.13 Stepping Through a Program 102
4.13.1 Executing the [Step In] Command 102
4.13.2 Executing the [Step Out] Command 104
4.13.3 Executing the [Step Over] Command 105
4.14 Forced Breaking of Program Executions 106
4.15 Resetting the Target MCU 106
4.16 Break Function 107
4.16.1 Software Break Function 107
4.16.2 On-Chip Break Function 113
4.17 Trace Functions 115
4.17.1 Displaying Trace Information by the Free Trace Function 116
4.17.2 Displaying Trace Information by the Trace Stop Function 118
4.17.3 Displaying Trace Information by the Conditional Trace Function 121
4.17.4 Statistics 122
4.17.5 Function Calls 126
4.18 Stack Trace Function 127
4.19 Performance Analysis Function 129
4.19.1 Time Of Specified Range Measurement 129
4.20 Profiling Function 132
4.21 Monitor Function 137
4.22 What Next? 139

Section 5 Software Specifications and Notes Specific to This Product 141
5.1 Supported Hardware 141
5.2 Target Platforms 141
5.3 Memory Map 142
5.4 Displaying and Modifying the Contents of Memory 143
5.4.1 Displaying and Modifying the Contents of Memory during Execution 143
5.4.2 Monitor Function 143
5.4.3 Parallel Access Function 144
5.5 Executing Your Program 144
5.5.1 Step Execution 144
5.6 Event Functions 145
5.6.1 Software Breakpoints 145
5.6.2 On-Chip Break 145
5.6.3 On-Emulator Break 146
5.7 Trace Functions 149
5.7.1 Displaying the Trace Information 149
5.7.2 Specifying Trace Acquisition Conditions 149
5.7.3 Searching for a Trace Record 149
5.7.4 Filtering Trace Records 149
5.8 Monitor Function 149
5.9 Performance Analysis Function 150
5.9.1 Errors 150
5.9.2 Notes 150
5.10 Profiling Function 150
5.11 RTOS Extension Function 151
5.11.1 Input Values for the [TASK ID writing area] Input Edit Box 151
5.11.2 Debugging Area for the E6000H Emulator 151
5.11.3 Note on Using the HI1000/4 Debugging Extension 151
5.12 Input Format 152
5.12.1 Entering Masks 152

Section 6 Error Messages 153
6.1 Error Messages of the Emulator 153
6.1.1 Error Messages at Emulator Initiation 153
6.1.2 Error Messages during Emulation 155

Appendix A Menus 157

Appendix B Command Lines 161

H8SX/1650 Hardware Part

Section 1 Overview

1.1 Notes on Usage

CAUTION

READ the following warnings before using the emulator product. Incorrect operation will damage the user system and the emulator product. The USER PROGRAM will be LOST.

1. Check all components with the component list after unpacking the emulator.
2. Never place heavy objects on the casing.
3. Observe the following conditions in the area where the emulator is to be used:
   - Make sure that the internal cooling fans on the sides of the emulator must be at least 20 cm (8") away from walls or other equipment.
   - Keep out of direct sunlight or heat. Refer to section 3.1, Environmental Conditions.
   - Use in an environment with constant temperature and humidity.
   - Protect the emulator from dust.
   - Avoid subjecting the emulator to excessive vibration. Refer to section 3.1, Environmental Conditions.
4. Protect the emulator from excessive impacts and stresses.
5. Before using the emulator's power supply, check its specifications such as power voltage and frequency.
6. When moving the emulator, take care not to subject it to strong vibration or mechanical shock.
7. After connecting the cable, check that it is connected correctly. For details, refer to section 2, Preparation before Use.
8. Supply power to the emulator and connected parts after connecting all cables. Cables must not be connected or removed while the power is on.
9. For details on differences between the target MCU and the emulator, refer to section 3.4, Support of the Target MCU.

1.2 Emulator Hardware Components

The emulator consists of an E6000H station and an evaluation chip board. By installing a user system interface board (option) on your host computer, the emulator can be connected in the same package as the device. PC interface (option) includes a PC interface board (PCI bus and PC card bus), a LAN adapter (connected with the network), and a USB adapter (connected with the USB interface). By connecting the emulator to the host computer via those interfaces, the High-performance Embedded Workshop can be used for debugging. For details on the PC interface boards (available for PCI bus and PC card bus specifications), the LAN adapter, and the USB adapter, refer to the relevant descriptive documents.

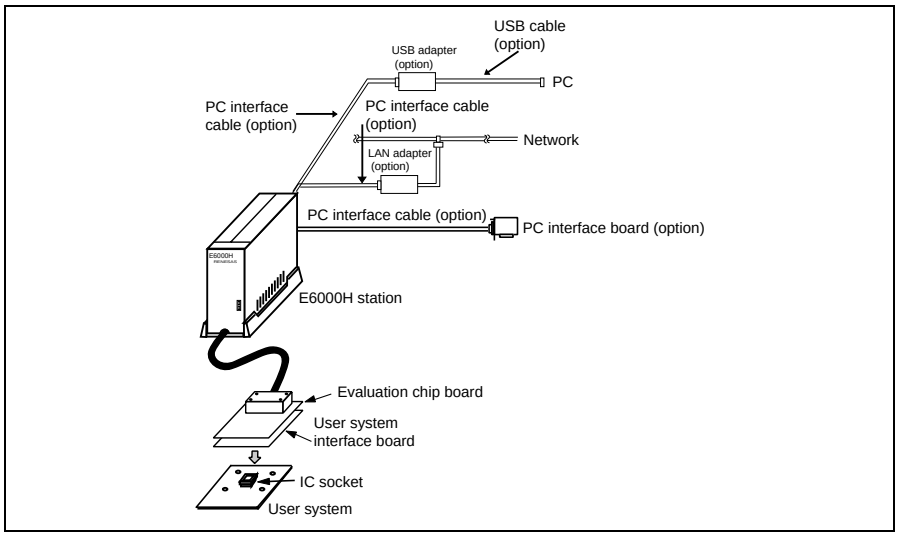


Figure 1.1 Emulator Hardware Components

1.2.1 E6000H Station Components (A Part of Photos may be Different from Real Appearances)

The names of the components on the front/rear panel of the E6000H station are listed below.

Front Panel:

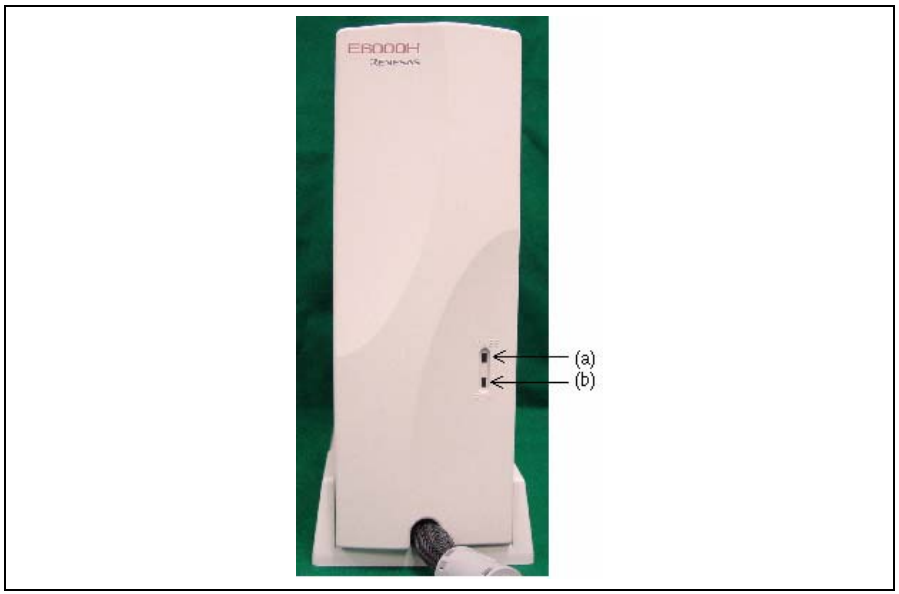


Figure 1.2 E6000H Station: Front Panel

(a): POWER lamp: Is lit up while the E6000H station is supplied with power.

(b): RUN lamp: Is lit up while the user program is running.

Rear Panel:

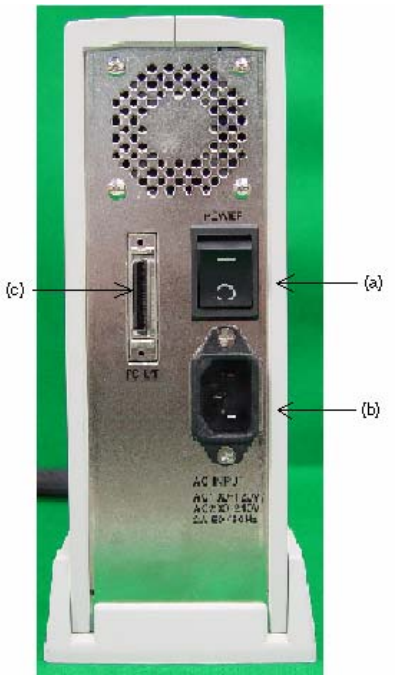


Figure 1.3 E6000H Station: Rear Panel

(a)	Power switch:	Turning this switch to I (input) supplies power to the emulator (E6000H station and evaluation chip board).
(b)	AC power connector:	For an AC 100-V to 240-V power supply.
(c)	PC interface cable connector:	For the PC interface cable that connects the host computer to the E6000H station. A PC interface board, PC card interface, LAN adapter, or USB adapter can be connected. Marked PC/IF.

1.2.2 Evaluation Chip Board Configuration

The names of the components on the evaluation chip board of the emulator are listed below.

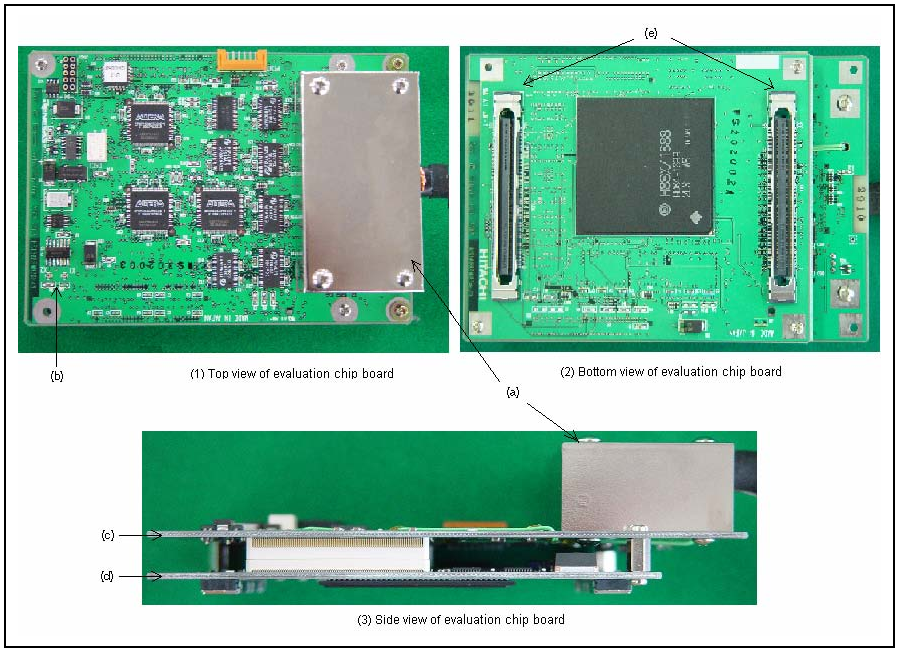


Figure 1.4 Evaluation Chip Board

(a) Station to evaluation chip board interface connector cover:	This is a cover for protecting the connector that connects the E6000H station to the evaluation chip board.
(b) Crystal oscillator terminals:	For installing a crystal oscillator to be used as an external clock source for the MCU.
(c) HS1650PWB20H board:	Connector to the trace cable is attached.
(d) HS1650PWB30H board:	An evaluation chip is installed and a dedicated connector to the user system interface board or user system is attached.
(e) User system interface board connector:	For connecting the user system interface board or user system.

Note: (a) to (e) listed above are referred to as evaluation chip board.

1.3 System Configuration

The emulator must be connected to a host computer via the selected PC interface board (PCI bus or PC card bus), LAN adapter, or USB adapter. For details on the PC interface boards (available for PCI bus and PC card bus specifications), the LAN adapter, and the USB adapter, refer to the relevant descriptive documents.

1.3.1 System Configuration Using Various Interfaces

(1) PC Interface Board

Figure 1.5 shows the configuration of a system in which the PC interface board is used. The emulator can be connected to a host computer via a PC interface board (option: PCI bus or PC card bus). Install the PC interface board to the expansion slot for the interface board in the host computer, and connect the interface cable supplied with the PC interface board to the emulator.

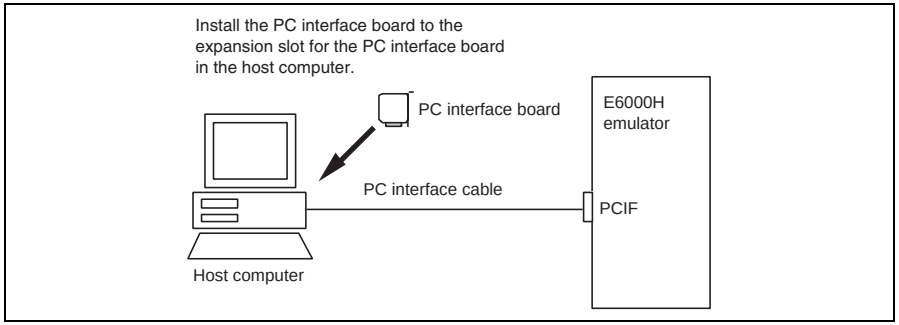


Figure 1.5 System Configuration Using a PC Interface Board

(2) LAN Adapter

Figure 1.6 shows the configuration of a system in which the LAN adapter is used. A LAN adapter can be used to connect the emulator to a host computer as a network.

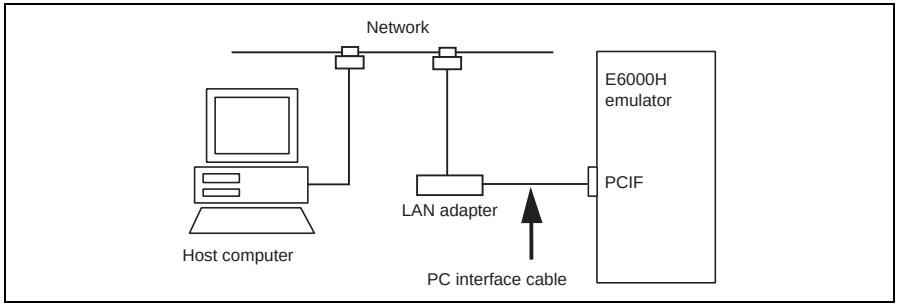


Figure 1.6 System Configuration Using a LAN Adapter

(3) USB Adapter

Figure 1.7 shows the configuration of a system in which the USB adapter is used. A USB adapter can be used to connect the emulator to a host computer with the USB interface.

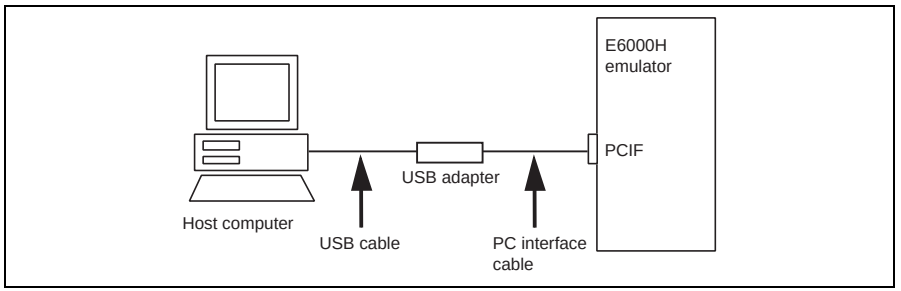


Figure 1.7 System Configuration Using a USB Adapter

Section 2 Preparation before Use

2.1 Description on Emulator Usage

This section describes the preparation before emulator usage. Figure 2.1 is a flowchart on preparation before use of the emulator.

CAUTION

Read this section and understand its contents before preparation.
Incorrect operation will damage the user system and the emulator.
The USER PROGRAM will be LOST.

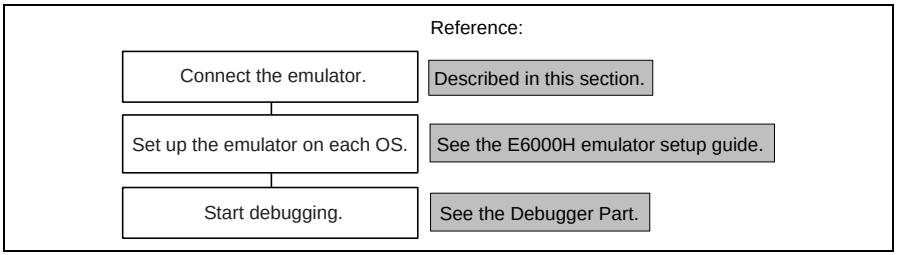


Figure 2.1 Emulator Preparation Flowchart

2.2 Emulator Connection

The following description covers connection of the emulator.

2.2.1 Connecting the Emulator to the User System

⚠WARNING

Always switch OFF the emulator and user system before connecting or disconnecting any CABLES. Failure to do so will result in a FIRE HAZARD and will damage the user system and the emulator or will result in PERSONAL INJURY. The USER PROGRAM will be LOST.

1. Check that the emulator power switch is turned off. Ensure that the power lamp on the right side of the E6000H station's front panel is not lit.
2. Remove the AC power cable of the E6000H station from the outlet (if the cable is connected to the outlet).
3. The emulator is connected to the user system by using the user system interface board or the user system interface connectors installed on the bottom of the E6000H evaluation chip board.

2.2.2 Connecting the User System Interface Board

⚠WARNING

Always switch OFF the emulator and user system and check pin numbers on the connectors and IC socket before connecting or disconnecting the USER SYSTEM INTERFACE BOARD. Connection with the power on or incorrect connection will damage the emulator, user system interface board, and user system, and result in a FIRE HAZARD.

For details on the method of connecting the user system interface board, refer to the descriptions of the user system interface boards for individual H8SX/1650 E6000H-series products.

2.2.3 Connection by the User System Interface Connectors

⚠WARNING

Always switch OFF the emulator and user system before connecting or disconnecting any CABLES. Failure to do so will result in a FIRE HAZARD, and will damage the user system or emulator or result in PERSONAL INJURY. Also, the USER PROGRAM will be LOST. For correct connection, check the location of pin 1.

Connect pin 1 on the user system connector to the connector installed at the bottom of the E6000H user system interface board. When connecting the connector, prevent the upper or lower side of the board from lifting off the connector. Alternately tighten the screws on both sides of the board.

Note: This evaluation chip board can only be used in combination with the specified dedicated connectors (WD-200P-VF85-N).

Install the connectors (WD-200P-VF85-N manufactured by Japan Aviation Electronics Industry, Ltd.) on the user system to connect the emulator. Figures 2.2 to 2.4 show connection using the dedicated connector, size restrictions for the installed components, and the location for mounting the connector in the user system, respectively.

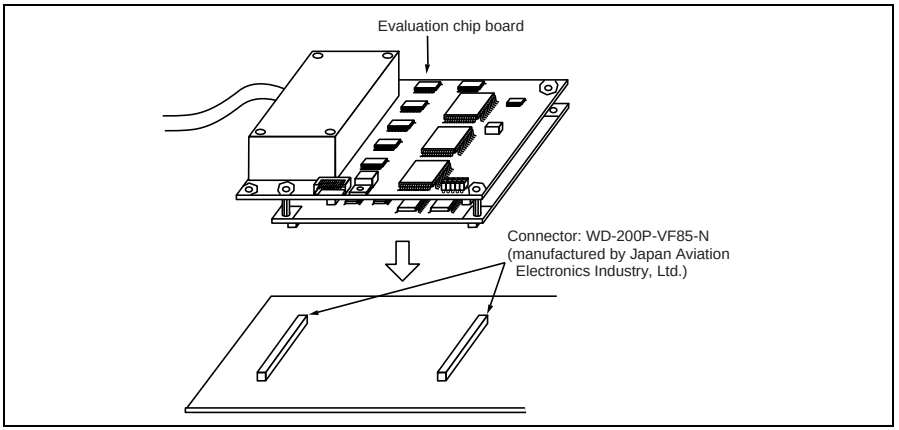


Figure 2.2 Connection Using the Dedicated Connectors

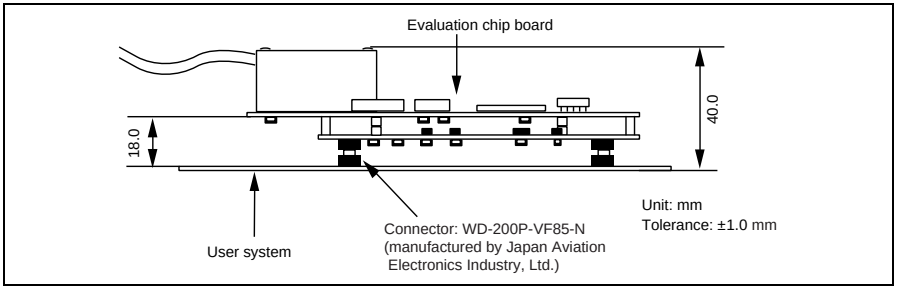


Figure 2.3 Size Restrictions for the Installed Components

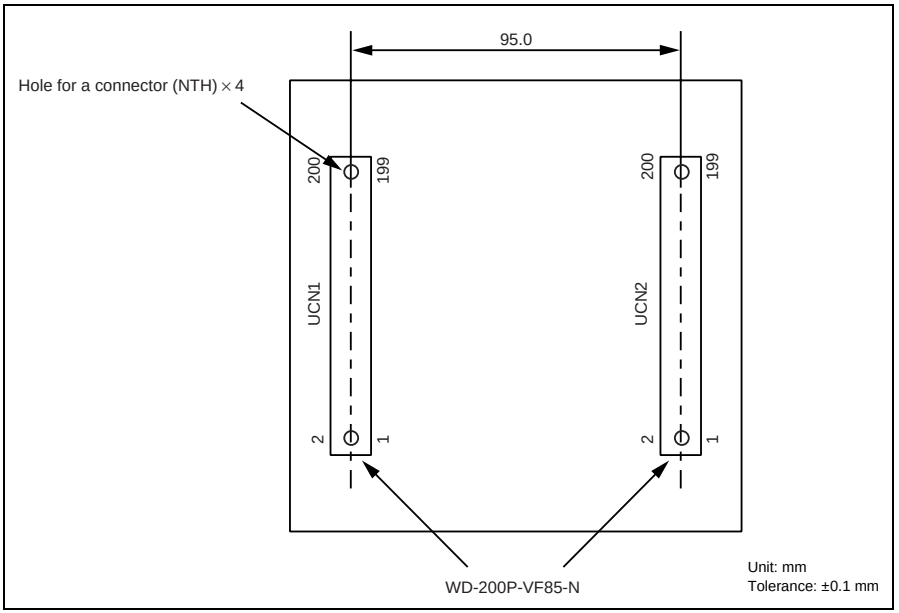


Figure 2.4 Location for Mounting the Connector in the User System

To design the foot pattern, refer to the catalog on WD-200P-VF85-N for dimensions.

2.2.4 Pin Arrangement on the User System Interface Connector

Table 2.1 lists the pin arrangement on the user system interface connector of HS1650EPH60H.

Table 2.1 Pin Arrangement on HS1650EPH60H

User I/F 1	Pin No.	Signal Name	User I/F 1	Pin No.	Signal Name
UCN1	1	VREF	UCN1	25	P64
	2	—		26	P65
	3	—		27	—
	4	Avss		28	—
	5	Avcc		29	VSS_8
	6	VSS_0		30	VSS_9
	7	—		31	—
	8	VSS_1		32	—
	9	—		33	—
	10	—		34	—
	11	—		35	VSS_10
	12	—		36	—
	13	—		37	—
	14	VSS_5		38	—
	15	RES		39	—
	16	STBY		40	VSS_11
	17	NMI		41	VSS_12
	18	—		42	P50
	19	VSS_6		43	P51
	20	P60		44	P52
	21	P61		45	P53
	22	P62		46	VSS_13
	23	P63		47	P54
	24	VSS_7		48	P55

Table 2.1 Pin Arrangement on HS1650EPH60H (cont)

User I/F 1	Pin No.	Signal Name	User I/F 1	Pin No.	Signal Name
UCN1	49	P56	UCN1	75	VSS_19
	50	P57		76	VSS_20
	51	VSS_14		77	—
	52	P10		78	—
	53	P11		79	—
	54	P12		80	—
	55	P13		81	VSS_21
	56	VSS_15		82	VSS_22
	57	P14		83	PD0
	58	P15		84	PD1
	59	P16		85	PD2
	60	P17		86	PD3
	61	VSS_16		87	VSS_23
	62	P20		88	VSS_24
	63	P21		89	PD4
	64	P22		90	PD5
	65	P23		91	PD6
	66	VSS_17		92	PD7
	67	P24		93	VSS_25
	68	P25		94	VSS_26
	69	P26		95	PE0
	70	P27		96	PE1
	71	VSS_18		97	PE2
	72	—		98	PE3
	73	—		99	VSS_27
	74	—		100	VSS_28

Table 2.1 Pin Arrangement on HS1650EPH60H (cont)

User I/F 1	Pin No.	Signal Name	User I/F 1	Pin No.	Signal Name
UCN1	101	PE4	UCN1	127	VSS_35
	102	PE5		128	PA0
	103	PE6		129	PA1
	104	PE7		130	PA2
	105	VSS_29		131	PA3
	106	VSS_30		132	PA4
	107	PF0		133	PA5
	108	PF1		134	PA6
	109	PF2		135	PA7
	110	PF3		136	VSS_36
	111	VSS_31		137	PB0
	112	VSS_32		138	PB1
	113	PF4		139	PB2
	114	PF5		140	PB3
	115	PF6		141	—
	116	PF7		142	—
	117	VSS_33		143	—
	118	VSS_34		144	—
	119	—		145	VSS_37
	120	—		146	—
	121	—		147	—
	122	—		148	—
	123	—		149	—
	124	—		150	—
	125	—		151	—
	126	—		152	—

Table 2.1 Pin Arrangement on HS1650EPH60H (cont)

User I/F 1	Pin No.	Signal Name	User I/F 1	Pin No.	Signal Name
UCN1	153	VSS_38	UCN1	177	VSS_46
	154	VSS_39		178	VSS_47
	155	PH0		179	—
	156	PH1		180	—
	157	PH2		181	—
	158	PH3		182	—
	159	VSS_40		183	VSS_48
	160	VSS_41		184	VSS_49
	161	PH4		185	—
	162	PH5		186	—
	163	PH6		187	—
	164	PH7		188	—
	165	VSS_42		189	VSS_50
	166	VSS_43		190	VSS_51
	167	PI0		191	—
	168	PI1		192	—
	169	PI2		193	—
	170	PI3		194	—
	171	VSS_44		195	VSS_52
	172	VSS_45		196	VSS_53
	173	PI4		197	—
	174	PI5		198	—
	175	PI6		199	—
	176	PI7		200	—

Table 2.1 Pin Arrangement on HS1650EPH60H (cont)

User I/F 2	Pin No.	Signal Name	User I/F 2	Pin No.	Signal Name
UCN2	1	—	UCN2	31	—
	2	—		32	—
	3	—		33	—
	4	—		34	VSS_59
	5	—		35	—
	6	—		36	—
	7	VSS_54		37	—
	8	P30		38	—
	9	P31		39	VSS_60
	10	P32		40	—
	11	P33		41	—
	12	VSS_55		42	—
	13	P34		43	—
	14	P35		44	VSS_61
	15	P36		45	—
	16	P37		46	—
	17	VSS_56		47	—
	18	—		48	—
	19	—		49	VSS_62
	20	—		50	—
	21	—		51	—
	22	—		52	—
	23	—		53	—
	24	VSS_57		54	VSS_63
	25	—		55	—
	26	—		56	—
	27	—		57	—
	28	—		58	—
	29	VSS_58		59	VSS_64
	30	—		60	—

Table 2.1 Pin Arrangement on HS1650EPH60H (cont)

User I/F 2	Pin No.	Signal Name	User I/F 2	Pin No.	Signal Name
UCN2	61	—	UCN2	90	—
	62	—		91	—
	63	—		92	—
	64	—		93	VSS_68
	65	—		94	—
	66	—		95	—
	67	—		96	—
	68	VSS_65		97	—
	69	—		98	WDTOVF
	70	—		99	—
	71	—		100	VSS_69
	72	—		101	—
	73	—		102	—
	74	—		103	—
	75	—		104	—
	76	—		105	VSS_70
	77	VSS_66		106	—
	78	—		107	—
	79	—		108	—
	80	—		109	VSS_71
	81	—		110	—
	82	VSS_67		111	—
	83	—		112	VSS_72
	84	—		113	—
	85	—		114	—
	86	—		115	—
	87	—		116	—
	88	—		117	—
	89	—		118	VSS_73

Table 2.1 Pin Arrangement on HS1650EPH60H (cont)

User I/F 2	Pin No.	Signal Name	User I/F 2	Pin No.	Signal Name
UCN2	119	—	UCN2	146	—
	120	—		147	VSS_79
	121	Vcc		148	—
	122	VSS_74		149	—
	123	—		150	—
	124	—		151	—
	125	—		152	VSS_80
	126	—		153	—
	127	VSS_75		154	—
	128	—		155	—
	129	—		156	—
	130	—		157	VSS_81
	131	—		158	—
	132	VSS_76		159	—
	133	—		160	—
	134	—		161	—
	135	—		162	VSS_82
	136	—		163	—
	137	VSS_77		164	—
	138	—		165	—
	139	—		166	—
	140	—		167	VSS_83
	141	—		168	—
	142	VSS_78		169	—
	143	—		170	—
	144	—		171	—
	145	—		172	VSS_84

Table 2.1 Pin Arrangement on HS1650EPH60H (cont)

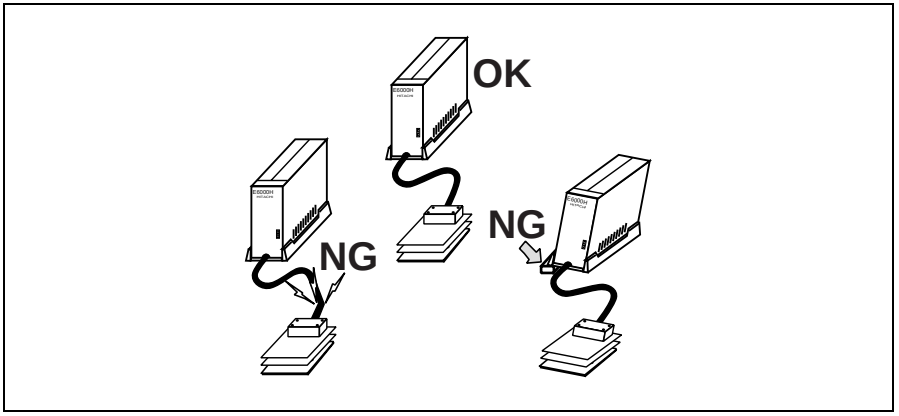
User I/F 2	Pin No.	Signal Name	User I/F 2	Pin No.	Signal Name
UCN2	173	—	UCN2	187	MD0
	174	—		188	MD1
	175	—		189	MD2
	176	—		190	—
	177	VSS_85		191	—
	178	—		192	—
	179	—		193	VSS_3
	180	—		194	VSS_2
	181	VSS_86		195	—
	182	—		196	EXTAL
	183	—		197	VSS_87
	184	—		198	—
	185	—		199	—
	186	—		200	—

Note: Do not connect anything to pins indicated as '—'. Failure to do so will damage the emulator or result in a fire hazard.

2.2.5 Precautions on Connecting the User System

When connecting the evaluation chip board to the user system, note the following:

1. Secure the E6000H station location.
   Place the E6000H station and evaluation chip board so that the trace cable is not bent or twisted, as shown below. A bent or twisted cable will impose stress on the user interface, leading to connection or contact failure. Make sure that the E6000H station is placed in a secure position so that it does not move and impose stress on the user interface while being used.



2. Make sure the power supply is off.
   Before connecting the evaluation chip board to the user system, check that the emulator and the user system are turned off.
3. Connect Vcc to the user system power.
   The emulator monitors and determines whether the user system is turned on or off by the following Vcc pins:
   (a) Connecting the dedicated connector for HS1650EPH60H: Pin UCN2-121
   (b) Connecting the user system interface board: Vcc: All Vcc pins

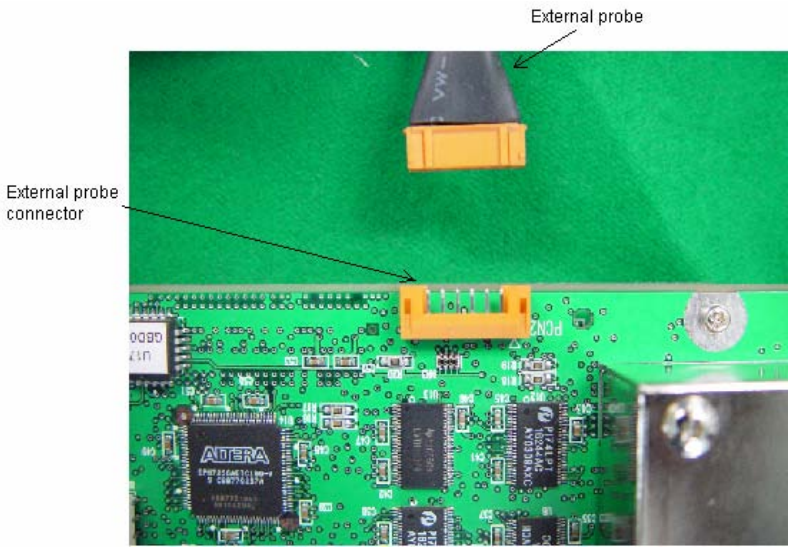
   Accordingly, after connecting the user system to the emulator, be sure to supply power to the Vcc pins. Otherwise, the emulator assumes that the user system is not connected.
   When the user system is connected, check that the power of the user system is supplied to these pins.

2.2.6 Connecting the External Probe (Not Supported in this Emulator)

CAUTION

Check the external probe direction and connect the external probe to the emulator station correctly. Incorrect connection will damage the probe or connector.

When an external probe is connected to the emulator probe connector on the E6000H evaluation chip board's rear panel, it enables external signal tracing and multibreak detection. Figure 2.5 shows the external probe connector.



Pin No.	Probe Name	Signal Name	Note
1	1	Probe input 0	Synchronous break input pin
2	2	Probe input 1	
3	3	Probe input 2	
4	4	Probe input 3	
5	5	GND	GND connection pin
6	6	Trigger output	Trigger mode output pin
7	7	GND	GND connection pin
8	8	RUN or break output	High-level output at RUNSTEP

Figure 2.5 External Probe Connector

2.2.7 Selecting the Clock

This emulator supports three types of clock for the MCUs listed in table 2.2: a crystal oscillator attached on the evaluation chip board, external clock input from the user system, and the emulator internal clock. The clock is specified with the [Configuration Properties] dialog box.

Table 2.2 Maximum Operating Frequency of the System Clock

MCU	Maximum Operating Frequency of the System Clock (ϕ)	Frequency Range of the Crystal Oscillator
H8SX/1650	35 MHz	8 to 18 MHz
H8SX/1657		
H8SX/1651	50 MHz	
H8SX/1653		
H8SX/1654		
H8SX/1663		
H8SX/1664		

Table 2.3 CLOCK Command

MCU	Mode Clock	CLOCK Command
H8SX/1653 H8SX/1654 H8SX/1663 H8SX/1664	MD_CLK_0	12 (Emulator clock: 12 MHz) 16 (Emulator clock: 16 MHz) Target (External clock: 8 to 18 MHz) Xtal (Crystal oscillator: 8 to 18 MHz)
	MD_CLK_1	16 (Emulator clock: 16 MHz) Target (External clock: 8 to 18 MHz) Xtal (Crystal oscillator: 8 to 18 MHz)
H8SX/1650 H8SX/1657 H8SX/1651	—	12 (Emulator clock: 12 MHz) 16 (Emulator clock: 16 MHz) Target (External clock: 8 to 18 MHz) Xtal (Crystal oscillator: 8 to 18 MHz)

Crystal Oscillator: A crystal oscillator is not supplied with the emulator. Prepare and use one that has the same frequency as that of the user system. When using a crystal oscillator as the MCU clock source, the frequency range must be as shown in table 2.2.

CAUTION

Always switch OFF the emulator and user system before connecting or disconnecting the CRYSTAL OSCILLATOR. Otherwise, the USER PROGRAM will be LOST.

Follow the procedure listed below to install the crystal oscillator:

1. Check that the emulator power switch is turned off. (Check that the power LED is not lit.)
2. Attach the crystal oscillator into the terminals on the evaluation chip board (figure 2.6).
3. Turn on the user system power and then the emulator power. The crystal oscillator will then be automatically set and started up. This function will allow the execution of the user program at the operating frequency of the user system even when the user system is not connected to the emulator.

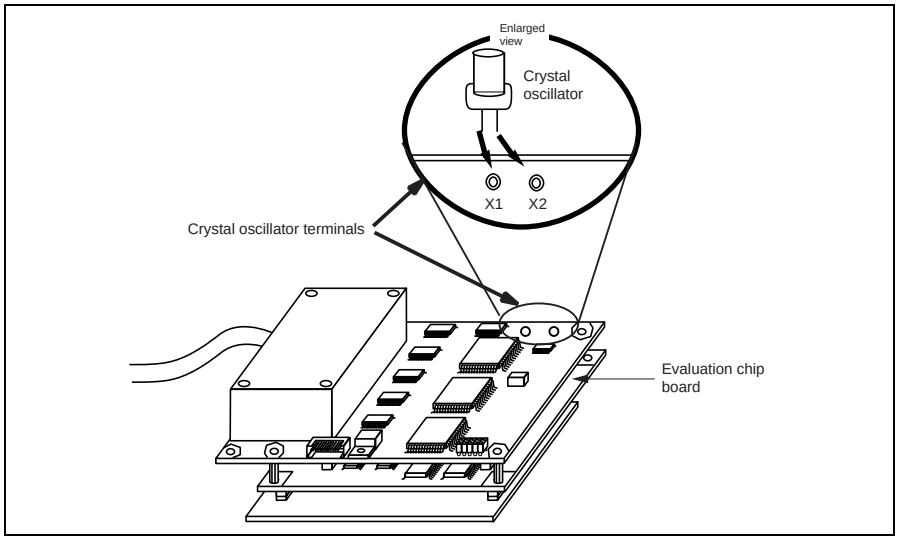


Figure 2.6 Installing the Crystal Oscillator

External Clock: Follow the procedure listed below to select the external clock.

1. Check that the emulator power switch is turned off. (Check that the power LED is not lit.)
2. Connect the user system interface cable to the user system and supply a clock signal through the EXTAL pin from the user system.
3. Turn on the user system power and then the emulator power. The external clock source will then be automatically specified.

Emulator Internal Clock: Specify 12.0 MHz or 16.0 MHz in the [Configuration] dialog box.

Reference:

When the emulator system program is initiated, the emulator automatically selects the MCU clock source according to the following priority:

1. User system's clock when an external clock is supplied from the user system
2. Crystal oscillator, if one is mounted on the evaluation chip board
3. Emulator-internal clock

2.2.8 Connecting the System Ground

CAUTION

Separate the frame ground from the signal ground at the user system. When the frame ground is connected to the signal ground and the emulator is then connected to the user system, the emulator will malfunction.

The emulator's signal ground is connected to the user system's signal ground via the evaluation chip board. In the E6000H station, the signal ground and frame ground are connected (figure 2.7). At the user system, connect the frame ground only; do not connect the signal ground to the frame ground.

If it is difficult to separate the frame ground from the signal ground in the user system, ground the frame to the same outlet as the 100-V to 240-V AC power supply of the emulator station (figure 2.8) so that the ground potentials become even.

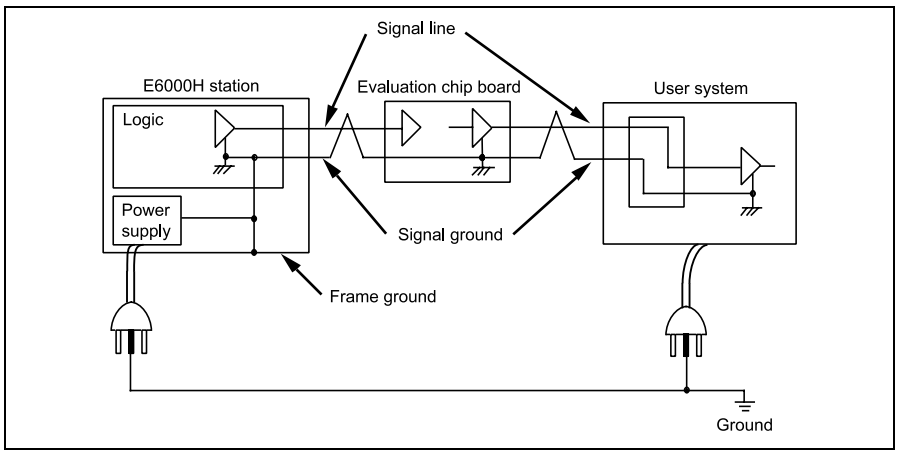


Figure 2.7 Connecting the System Ground

WARNING

Always switch OFF the emulator and user system before connecting or disconnecting any CABLES. Failure to do so will result in a FIRE HAZARD and will damage the user system and the emulator or will result in PERSONAL INJURY. The USER PROGRAM will be LOST.

The user system must be connected to an appropriate ground so as to minimize noise and the adverse effects of ground loops. When connecting the evaluation chip board and the user system, confirm that the ground pins of the evaluation chip board are firmly connected to the user system's ground.

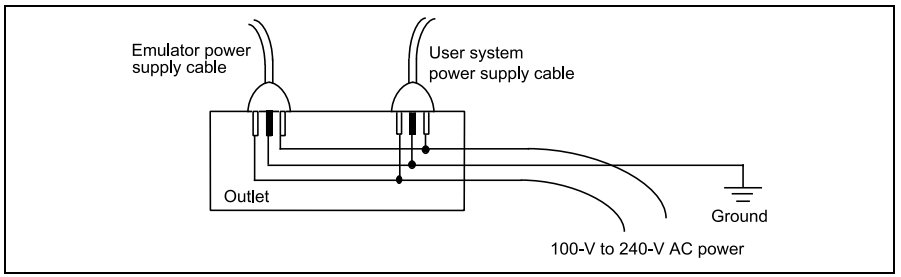


Figure 2.8 Connecting the Frame Ground

2.2.9 PC Interface Board Specifications

For details on the PC interface board, LAN adapter, or USB adapter, refer to their description notes.

Section 3 Hardware Specifications

3.1 Environmental Conditions

CAUTION

Observe the conditions listed in table 3.1 when using the emulator. The following environmental conditions must be satisfied, otherwise the user system and the emulator will not operate normally. The USER PROGRAM will be LOST.

Table 3.1 Environmental Conditions

Item	Specifications	
Temperature	Operating:	+10 to +35°C
	Storage:	−10 to +50°C
Humidity	Operating:	35 to 80% RH, no condensation
	Storage:	35 to 80% RH, no condensation
Vibration	Operating:	2.45 m/s^2 max.
	Storage:	4.9 m/s^2 max.
	Transportation:	14.7 m/s^2 max.
AC input power	Voltage:	100 V to 240 V AC
	Frequency:	50/60 Hz
	Power consumption:	75 W
Ambient gases	There must be no corrosive gases present.	

3.2 Emulator External Dimensions and Mass

Figures 3.1 shows the external dimensions and mass of the E6000H emulator.

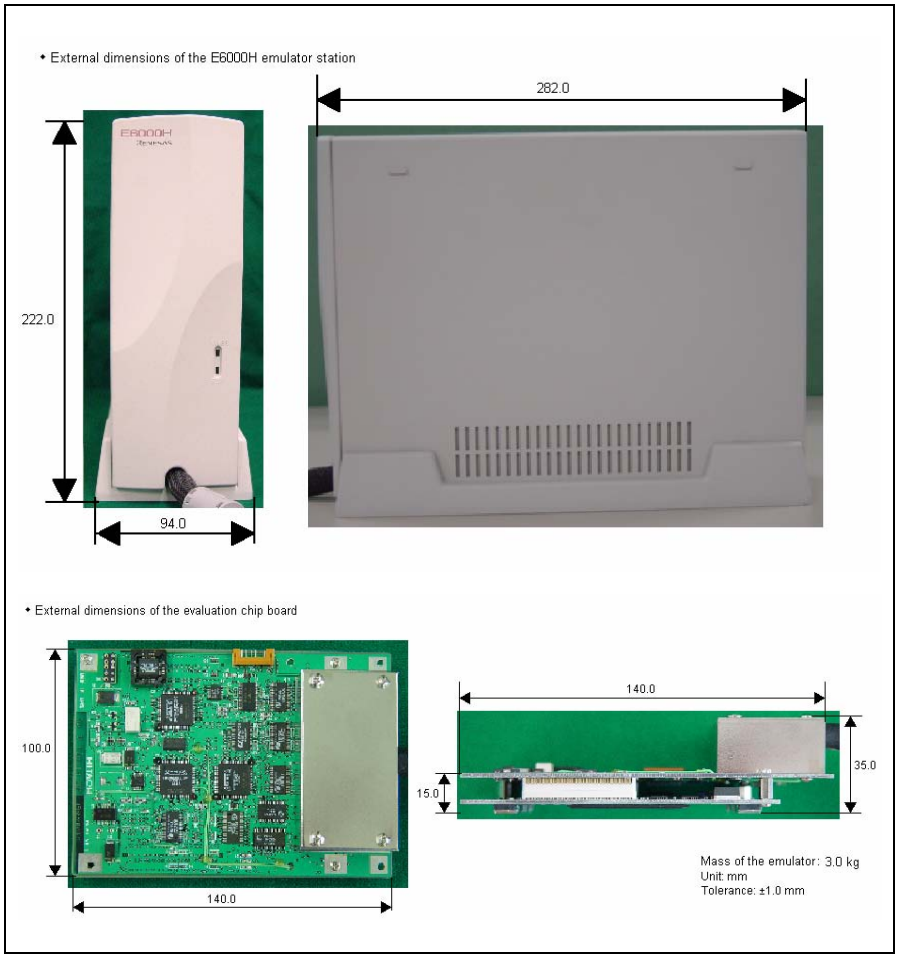


Figure 3.1 External Dimensions and Mass of the Emulator

3.3 User System Interface Circuit

3.3.1 User System Interface Circuit

The circuits that interface the evaluation chip in the emulator to the user system include buffers and resistors. When connecting the emulator to a user system, adjust the user system hardware compensating for FANIN, FANOUT, and propagation delays.

The AC timing values when using the emulator are shown in table 3.2.

Note: The values with the emulator connected, in table 3.2, are measurements for reference and are not guaranteed values.

Table 3.2 Bus Timing when Using the Emulator (Bus Clock of the H8SX/1600-series MCU: 35.0 MHz)

	MCU Specifications (ns)		Values with Emulator Connected (ns)	
Item	**Min**	**Max**	**Min**	**Max**
tRDS1	15	—	16	—

The basic bus cycle (software wait) is shown in figure 3.2. The user system interface circuits connected to the user system are shown in figure 3.3.

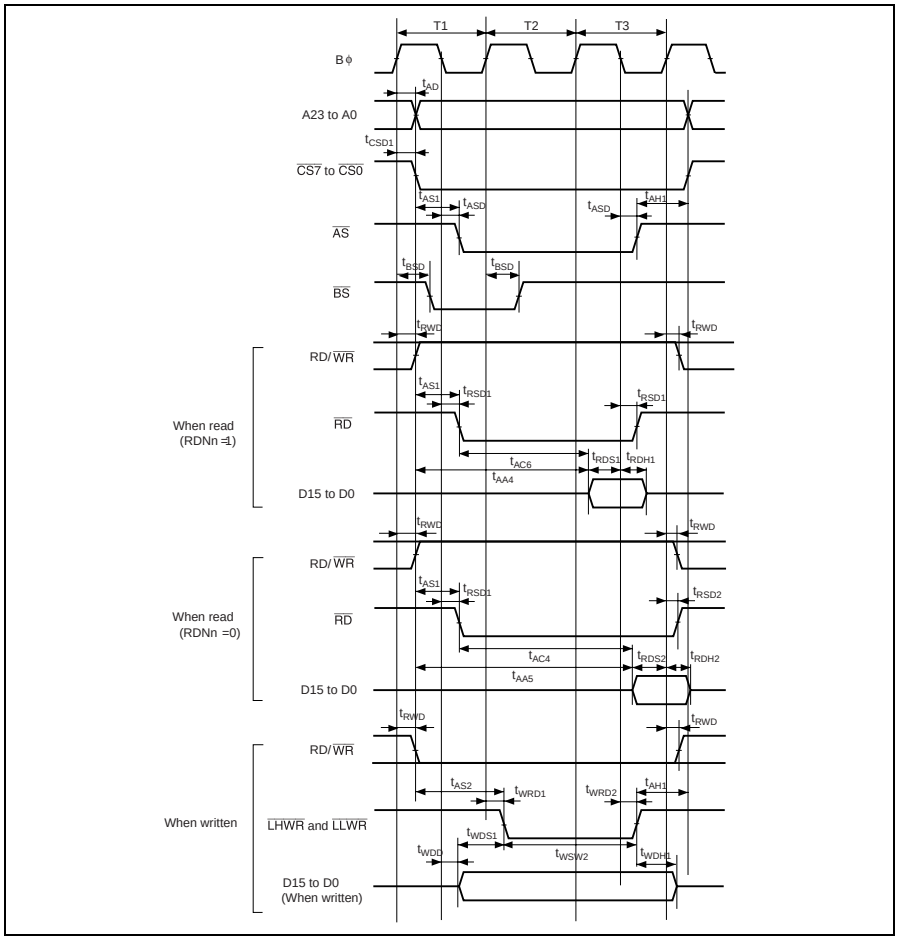


Figure 3.2 Basic Bus Cycle (Software Wait)

The delay time is generated on the timing of the _RES and _NMI signals when they are input to the evaluation chip from the user system, as shown in table 3.3, because this connection for those signals is via logic circuit on the evaluation chip board.

Table 3.3 Delay Time for Signal Connected via the Evaluation Chip Board

Signal Name	Delay Time (ns)
_RES	18
_NMI	18
_STBY	16

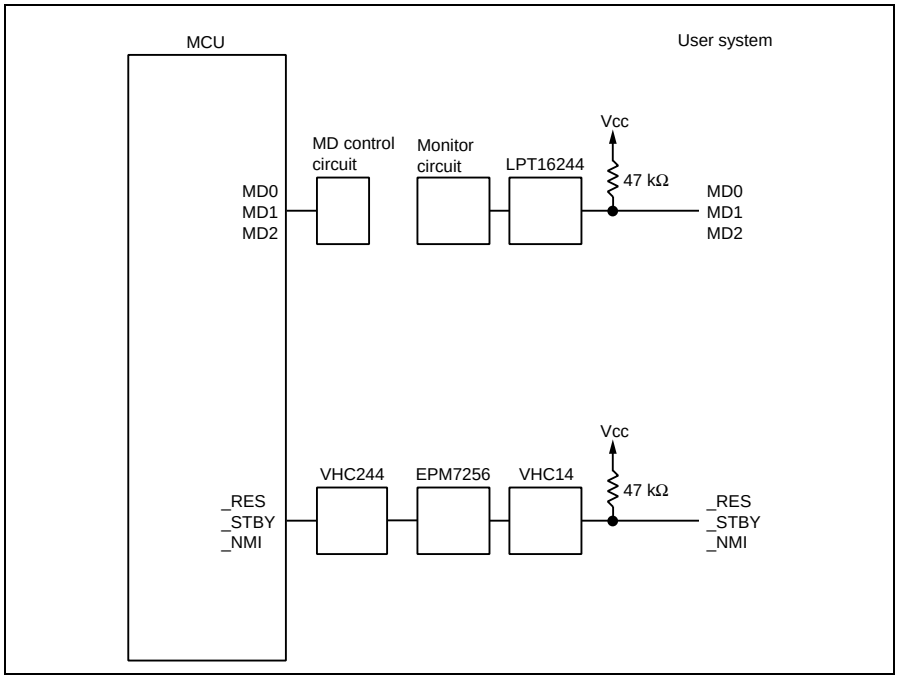


Figure 3.3 User System Interface Circuits (1)

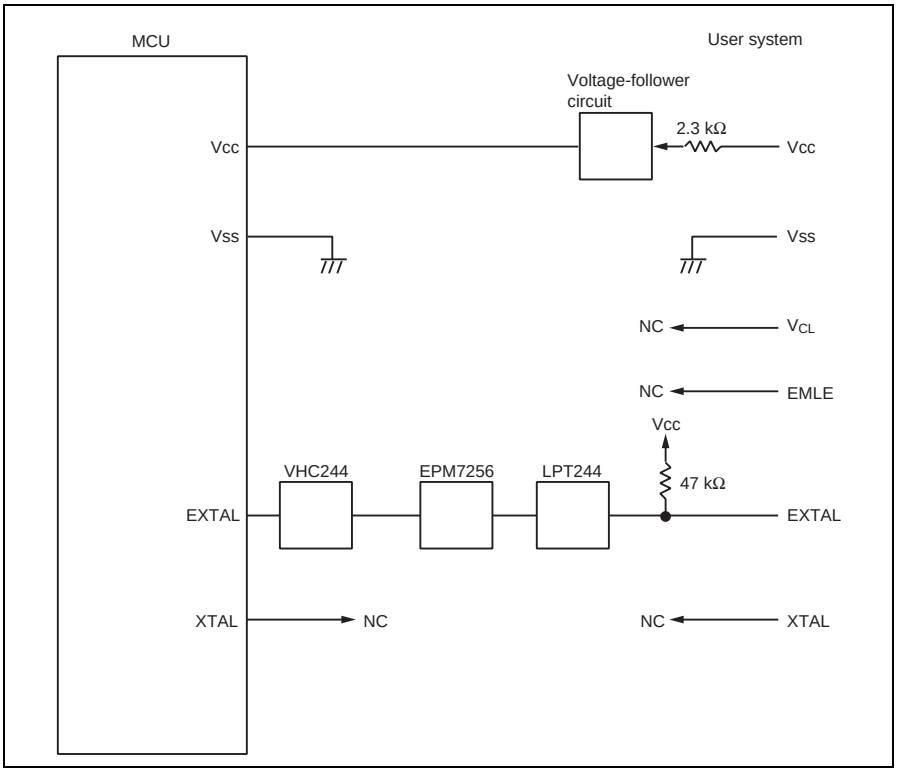


Figure 3.3 User System Interface Circuits (2)

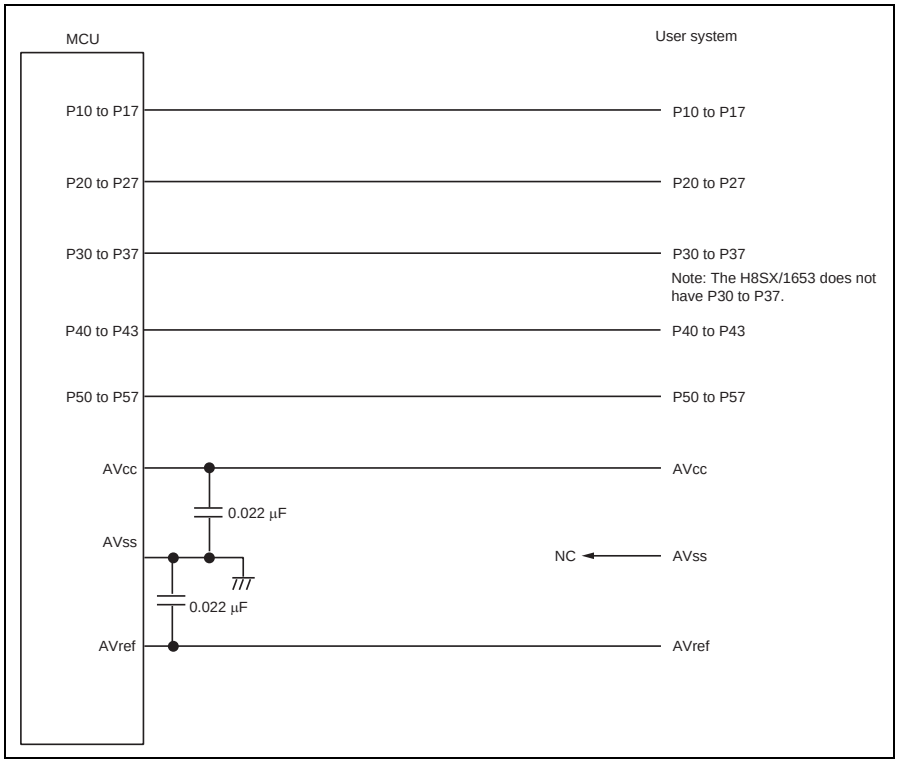


Figure 3.3 User System Interface Circuits (3)

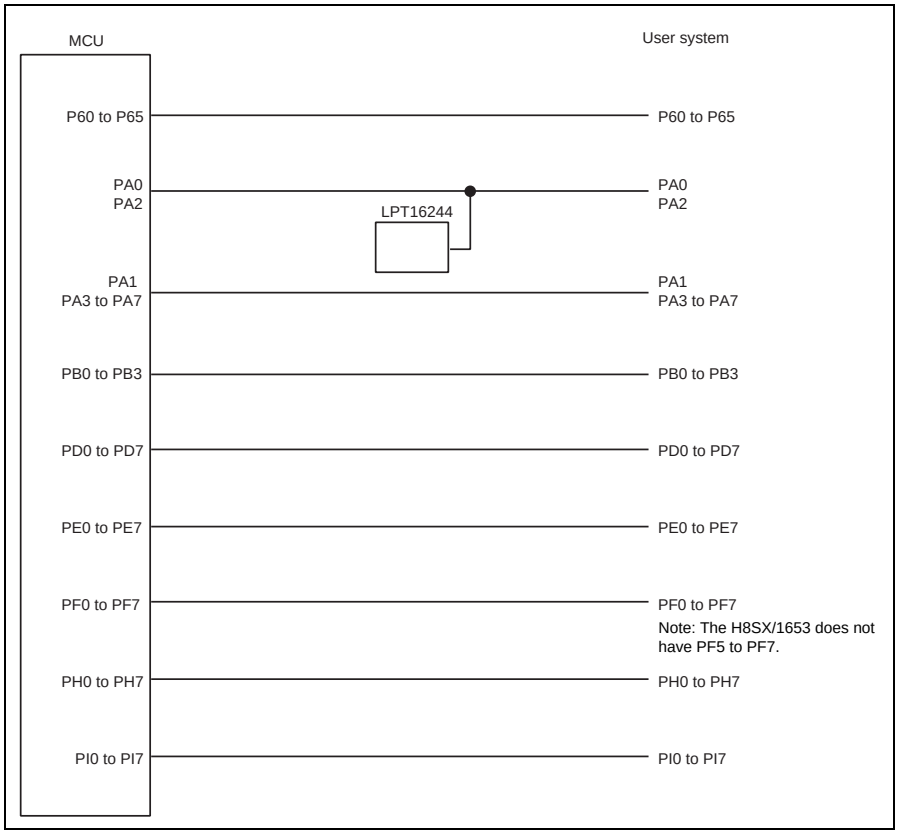


Figure 3.3 User System Interface Circuits (4)

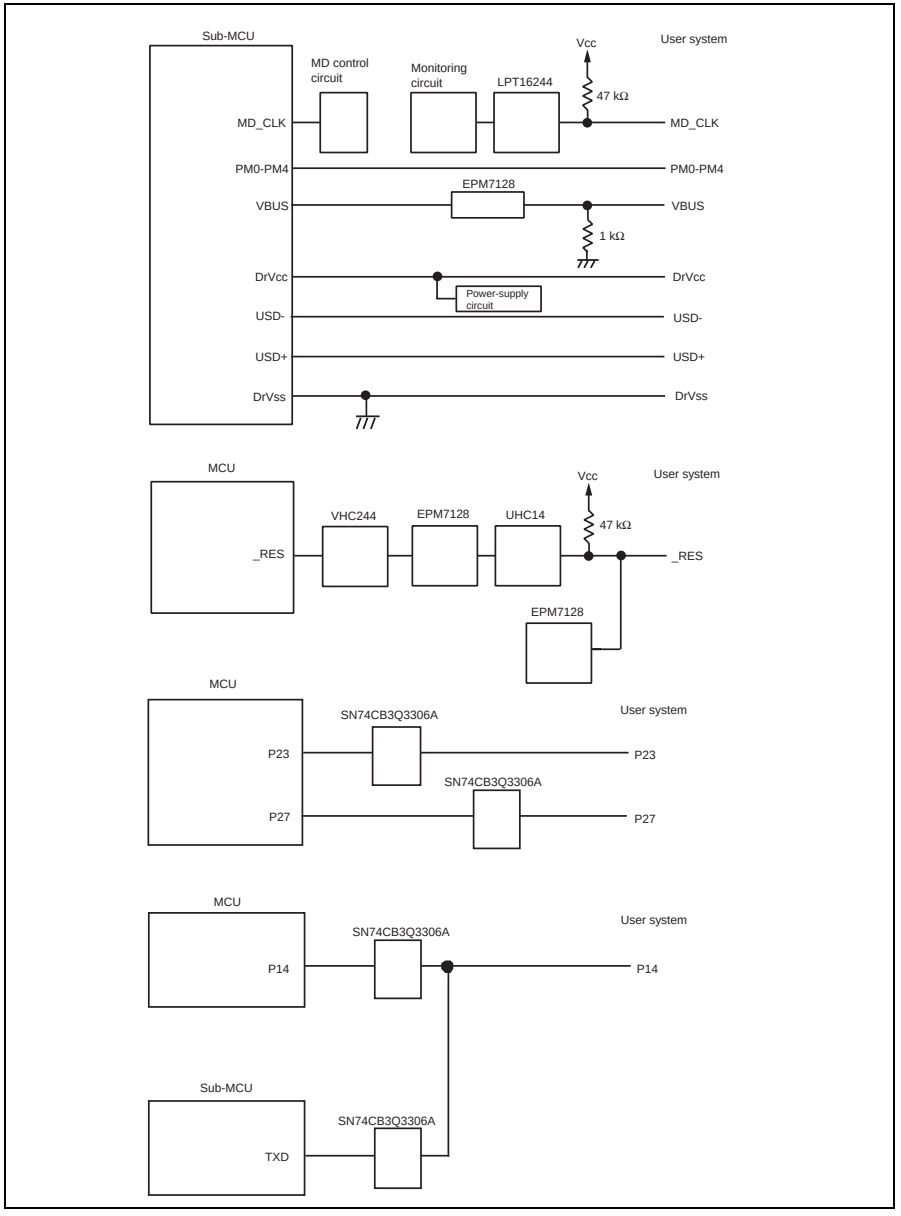


Figure 3.4 User System Interface Circuits (with HS1653ECN61H Connected)

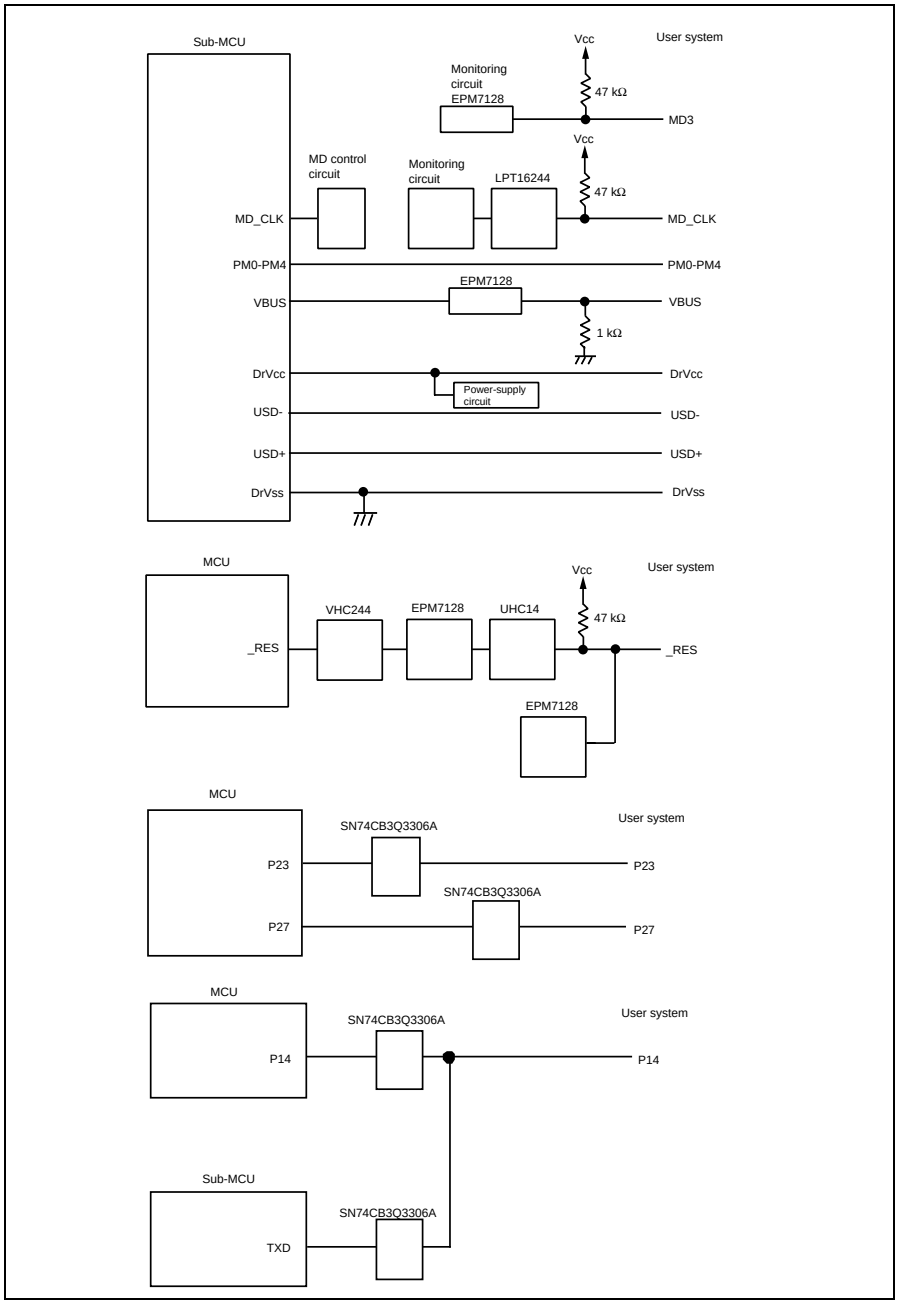


Figure 3.5 User System Interface Circuits (with HS1664ECH61H Connected)

Note: The power-supply circuits shown in figures 3.4 and 3.5 are turned on/off by the setting of the SW1 jumper pin on the user system interface board (HS1653ECN61H or HS1664ECH61H). Ensure that the jumper pin is inserted to [ON] when connecting the emulator (with the user system interface board attached) to the user system or supplying power to DrVCC. Otherwise the emulator product, user system interface board, and user system will be damaged.

3.4 Support of the Target MCU

3.4.1 Memory Space

The architecture of the H8SX/1600-series MCU allows for a 16-Mbyte memory space.

(1) Internal I/O Area

If an attempt is made to access the internal I/O area, the internal I/O area in the MCU installed in the emulator is accessed. To break user program execution when the on-chip I/O area is written to or accessed, use the hardware break or internal break.

(2) Internal RAM Area

If an attempt is made to access the internal RAM area, the internal RAM area in the MCU installed in the emulator is accessed. To break user program execution when the internal RAM area is written to or accessed, use the hardware break or internal break.

(3) External Memory Area

The MCU's external memory area can be set with all memory attributes that the emulator supports.

The emulator includes the 4-Mbyte emulation memory as the external memory area. Up to four blocks (one block is a maximum of 1 Mbyte) can be allocated to the emulation memory. To access this emulation memory, set wait-state cycles with the bus controller.

3.4.2 Power-Down Modes (Sleep, Software Standby, and Hardware Standby)

For reduced power consumption, the H8SX/1600-series MCU has sleep, software standby, and hardware standby modes.

(1) Sleep and Software Standby Modes

- Break

  The MCU can be taken out of the sleep and software standby modes either in the normal ways or through satisfaction of a break condition (forced break). When restarting after a break, the user program will restart at the instruction following the SLEEP instruction.

- Trace

  Trace information is not acquired in these modes.

- Memory access with emulator functions

  For information on displaying and modifying the contents of memory in the sleep and software standby modes, refer to section 5.4, Displaying and Modifying the Contents of Memory, in the Debugger Part.

3.4.3 Interrupts

During execution and step execution, the user can interrupt the H8SX/1600-series MCU.

Interrupt sources are retained while emulation is halted (break mode). In such cases, interrupt processing commences immediately after emulation is restarted.

3.4.4 Control Input Signals (/RES, /NMI, /BREQ, /WAIT, and /STBY)

The H8SX/1600-series MCU control input signals are /RES, /NMI, /BREQ, /WAIT, and /STBY. The /RES, /NMI, and /STBY signals are only valid when emulation was started with normal program execution (i.e., they are invalid when emulation was started with step execution). The /BREQ and /WAIT signals are valid during emulation with the display and modification of memory contents, execution, and step execution. While emulation is being halted (break), the input of the /RES or /NMI signal to the MCU is not possible.

The input of the /RES, /NMI, /STBY, /BREQ, or /WAIT signal during execution or step execution can be disabled by a setting in the [Configuration] dialog box.

3.4.5 Watchdog Timer (WDT)

When emulation is suspended (i.e. by a break), counting up by the WDT timer counter (TCNT) is also suspended. Counting resumes when emulation is resumed (user mode).

During break mode, a prescaler, which supplies a clock to TCNT, operates continuously. Since the prescaler might be in different phases before and after emulation goes through a period in the break mode, a break can change the period before the WDT overflows by ±1 cycle of the prescaler's clock.

3.4.6 A/D Converter

As well as analog input pins, the A/D converter has AVcc, AVss, Avref, and /ADTRG pins. As the A/D converter operates with an independent power supply, connect AVcc (the power supply pin) to the A/D power supply on the user system.

Notes: 1. When the A/D converter is not in use, connect AVcc to Vcc.
2. As the user system interface cable, wiring on the printed circuit board, and protective circuits are connected between the user system and the evaluation chip on the evaluation chip board, the precision of conversion is lower than that of the H8SX/1600-series MCU.

3.4.7 Emulator State and Internal Modules

Operation of some internal modules depends on the emulator's state. Table 3.4 shows the relation between the emulator's state and operation of the internal modules.

Table 3.4 Emulator's State and Operation of Internal Modules

Internal Module	Operation while Emulation is Halted (Break)	Operation During Emulation (Execution or Step Execution)
DMAC (DMA controller)	Yes	Yes
EXDMAC (EXDMAC controller)	Yes	Yes
DTC (data transfer controller)	Yes	Yes
TPU (16-bit timer pulse unit)	Yes	Yes
PPG (programmable pulse generator)	Yes	Yes
MTU (multi-function timer pulse unit)	Yes	Yes
TMR (8-bit timer)	Yes	Yes
WDT (watchdog timer)	No	Yes
SCI (serial communication interface)	Yes	Yes
SCIF (serial communication interface with FIFO)	Yes	Yes
IIC2 (IIC bus interface 2)	Yes	Yes
A/D converter	Yes	Yes
D/A converter	Yes	Yes
I/O port	Yes	Yes
H-UDI (user debugging interface)	Not available*	Not available*

Note: The user cannot use the H-UDI because it is being used by the emulator.

3.4.8 Differences in Values of Registers

Note that certain general and control registers of both emulators are initialized whenever the system is activated or the evaluation chip is reset by a command.

Table 3.5 Initial Values of Registers in the H8SX/1600-series MCU and the Emulator

	Emulator		
Register Name	**Power On**	**Reset (Reset CPU)**	**H8SX/1600-series MCU (Power-On Reset)**
PC	PC value indicated by the power-on reset vector	PC value indicated by the power-on reset vector	Undefined
ER0 to ER7	H'00000000	Value before the reset	Undefined
CCR	B'1XXXXXXX	B'1XXXXXXX	B'1XXXXXXX
EXR	B'01111111	Value before the reset	B'01111111
MACH	H'00000000	Value before the reset	Undefined
MACL	H'00000000	Value before the reset	Undefined
VBR	H'00000000	Value before the reset	H'00000000
SBR	H'FFFFFF00	Value before the reset	H'FFFFFF00

Note: X indicates an undefined bit.

3.4.9 Emulation Memory

The total size of emulation memory installed in this emulator is 4 Mbytes. This can be used as four 1-Mbyte blocks at maximum. To allocate memory, the start address must be aligned to a 1-Mbyte boundary. The emulation memory and user memory cannot coexist on a single address.

Emulation memory must be allocated to a 3-state access space. The number of access-state cycles for the emulation memory is three plus wait-state cycles. Set up the bus-state controller so that no wait-state cycle will be inserted for access to such areas.

3.5 Notes Specific to the H8SX/1650 E6000H Emulator

3.5.1 Custom Device Function

The maximum value selectable for the on-chip ROM with the custom device functions of the H8SX/1650 E6000H emulator is 2 Mbytes. Do not specify a value greater than 2 Mbytes.

3.5.2 Exception Processing of Sleep Instructions

The emulator does not support exception processing of sleep instructions. Even if the SLPIE bit of the standby control register (SBYCR) is set to 1, the emulator enters the power-down mode and no exception processing of sleep instructions occurs.

MCUs that apply: H8SX/1653, H8SX/1654, H8SX/1651, H8SX/1663, and H8SX/1664

3.5.3 Subclock Operation

This emulator does not support subclock operation. Thus the following function and register cannot be used:

- 32k timer
- Subclock control register (SUBCKCR) of the clock pulse generator

MCUs that apply: H8SX/1663 and H8SX/1664

Section 4 Diagnostic Test Procedure

This section describes the diagnostic test procedure using the E6000H diagnostic program.

4.1 System Set-Up for Diagnostic Program Execution

To execute the diagnostic program, use the following hardware; do not connect the user system interface board and user system.

- E6000H (HS1650EPH60H)
- Host computer
- The E6000 PC interface board which will be one of the following boards:
  PCI bus interface board (HS6000EIC01H or HS6000EIC02H)
  PC card interface (HS6000EIP01H)
  LAN adapter (HS6000ELN01H)
  USB adapter (HS6000EIU01H or HS6000EIU02H)

1. Install the E6000 PC interface board in the host computer and connect the supplied PC interface cable to the board.
2. Connect the PC interface cable to the emulator.
3. Connect the supplied AC power cable to the emulator.
4. Boot the host computer up and enter the command input wait state of DOS (the DOS prompt) (Windows® 98SE or Windows® Me) or the command prompt (Windows NT®, Windows® 2000, or Windows® XP). If the prompt window is not set up to be displayed over the whole screen, press the [Alt + Enter] key chord to switch the mode. To return the prompt to a window, press the [Alt + Enter] key. This key chord works in the same way regardless of the OS being used.

Note: If the display of the MS-DOS prompt does not switch from full-screen to windowed or vice-versa in response to the [Alt + Enter] key chord, mark the [Alt+Enter] check box in [Windows shortcut keys] of the [Misc] page on the [MS-DOS Prompt Properties] dialog box and click the [Apply] button to update the settings.

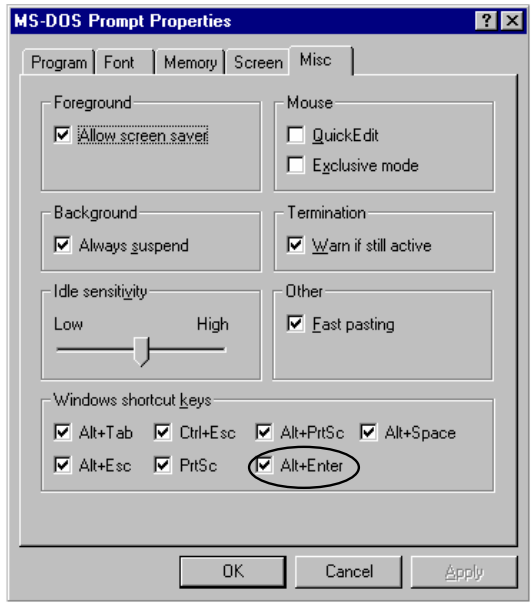


Figure 4.1 [Misc] Page

5. Turn on the E6000H emulator switch.

Note: To execute the diagnostic program, firstly turn on the power of the emulator. The diagnostic program checks the initial state of the hardware. Therefore, after turning on the power, do not activate the High-performance Embedded Workshop before executing the diagnostic program.

4.2 Test Item of the Diagnostic Program

Table 4.1 shows the test items of this diagnostic program.

Table 4.1 Test Items of the Diagnostic Program

Test No.	Test Item	Description
1	Main Board Access	Register test in the E6000H main board
2	Emulation Board Access	Register test in the E6000H emulation board
3	Evaluation Board Access	Register test in the E6000H evaluation chip board
4	Basic Function	Test of the basic functions
5	GO to BREAK Time Measurement	Test of the execution time measurement function
6	Emulation Monitor	Test of the emulation monitor
7	G/A Break Function	Test of the G/A break function
8	G/A Performance Analysis Function	Test of the G/A performance measurement function
9	G/A Monitor Function	Test of the G/A monitoring function
10	G/A Parallel RAM Monitor	Test of the G/A parallel RAM monitoring function
11	G/A Trace Function	Test of the G/A trace function
12	Combination	Test of several functions in combination
13	Parallel Access	Test of the parallel access function

4.3 Diagnostic Test Procedure Using the Diagnostic Program

Insert the CD-R (HS1650EPH60SR supplied with the emulator) into the CD-ROM drive of the host computer, use the command prompt to change the current directory to <Drive>:\Diag\1650, and enter the command below that corresponds to the PC interface board used to initiate the diagnostic program:

1. PCI bus interface board (HS6000EIC01H or HS6000EIC02H)
   > TM1650 –PCI (Enter)

2. PC card interface (HS6000EIP01H)
   > TM1650 –PCCD (Enter)

3. LAN adapter (HS6000ELN01H)
   > TM1650 –ELN (Enter)

4. USB adapter (HS6000EIU01H or HS6000EIU02H)
   > TM1650 –USB (Enter)

The High-performance Embedded Workshop must be installed before the test program is executed.

Be sure to initiate the diagnostic program from <Drive>:\Diag\1650. Do not initiate it from a directory other than <Drive>:\Diag\1650, such as > <Drive>:\Diag\TM1650 –PCI (Enter). If the diagnostic program is initiated when the current directory is not <Drive>:\Diag\1650, the diagnostic program will not operate correctly.

When –S is added to the command line, as in > TM1650 –PCI –S (Enter), steps 1 to 13 will be repeatedly executed. To stop execution in this case, enter Q.

Notes: 1. <Drive> is the drive name of the CD-ROM drive.
2. Do not remove the CD-R from the CD-ROM drive during execution of the test program.

The following messages are displayed during the test. There are 13 steps in this test.

Message	Description
E6000H H8SX/1650 Emulator Tests V*.* Copyright (c) 2003 Renesas Technology Corp.	Test program start message. x.x shows the version number.
Loading driver .....OK (Use PCI)	Shows that the PC interface board is correctly installed in the host computer.
Initializing driver .....OK	
Searching for interface card .....OK	
Checking emulator is connected .....OK	Shows that the E6000H emulator is correctly connected to the host computer.
Emulator board information: Main board ID: H'* Emulation board ID: H'***	Shows the ID number of the E6000H emulator.
Normal started at WWW MMM DD hh:mm:ss YYYY	Shows the time when the diagnostic program was started up.

```
******* NORMAL TEST - Press 'Q' to stop *******          (COUNT=0001)
  1. Main Board Access
     01) Registers Initial Value Check .....OK
     02) Registers Write/Verify .....OK
     03) DPRAM Address Decode Test .....SKIP
     04) DPRAM Marching Test .....SKIP
     05) Trace Memory Address Decode Test .....OK
     06) Trace Memory Marching Test .....OK
     07) G/A Registers Initial Value Check .....OK
     08) G/A Registers Write/Verify .....OK
  2. Emulation Board Access
     01) Registers Initial Value Check .....OK
     02) Registers Write/Verify .....OK
     03) RESERVED
     04) RESERVED
     05) RESERVED
     06) RESERVED
     07) RESERVED
     08) Firmware RAM Address Decode Test .....SKIP
     09) Firmware RAM Marching Test .....SKIP
     10) MAP Control RAM Address Decode Test .....OK
     11) MAP Control RAM Marching .....OK
  3. Evaluation Board Access
     01) Registers Initial Value Check .....OK
     02) Registers Write/Verify .....OK
     03) H-UDI IDCODE Check .....OK
     04) Firmware BOOT .....OK
     05) Configuration Set .....OK
  4. Basic Function
     01) GO to BREAK .....OK
     02) RESET GO .....OK
     03) STEP .....OK
     04) KEYBREAK .....OK
     05) BRKCONT .....OK
     06) Internal ROM Address Decode Test .....OK
     07) Internal ROM Marching Test .....OK
     08) Internal RAM Address Decode Test .....OK
     09) Internal RAM Marching Test .....OK
     10) Emulation RAM Address Decode Test .....OK
     11) Emulation RAM Marching Test .....OK
     12) MAP Break .....OK
  5. GO to BREAK Time Measurement
     01) Counter Test Mode (EMU 12MHz MPU 12MHz Sampling 20ns) ..OK
```

```
     02) EMU 12Mhz MPU 12Mhz Sampling 20ns .....................OK
     03) EMU 12Mhz MPU 12Mhz Sampling 1.6us .....................OK
     04) EMU 12Mhz MPU 12Mhz Sampling 52us .....................OK
     05) EMU 12Mhz MPU 12Mhz Sampling MPU .....................OK
     06) EMU 12Mhz MPU 12Mhz Sampling MPU/2 .....................OK
     07) EMU 12Mhz MPU 12Mhz Sampling MPU/4 .....................OK
     08) EMU 12Mhz MPU 12Mhz Sampling MPU/8 .....................OK
     09) EMU 12Mhz MPU 48Mhz Sampling 20ns .....................OK
     10) EMU  4Mhz MPU 32Mhz Sampling 20ns .....................OK
     11) EMU 35.7Mhz MPU 35.7Mhz Sampling 20ns ..................OK
     12) EMU 35.9Mhz MPU 35.9Mhz Sampling 20ns ..................OK
     13) EMU f1Mhz MPU f2Mhz Sampling 20ns ......................SKIP
  6. Emulation Monitor
     01) RESERVED
     02) RESERVED
     03) ASEST ..................................................OK
  7. G/A Break Function
     01) Address Condition ......................................OK
     02) Data Condition .........................................OK
     03) Control Signal Condition (ASEDSHH/HL/LH/HL) ............OK
     04) Function Code Condition (ASEST3-0) .....................OK
     05) RESERVED
     06) Control Signal Condition (CB22-19) .....................OK
     07) Control Signal Condition (CB18-16) .....................OK
  8. G/A Performance Analysis Function
     01) Time Measurement (20ns Sampling) .......................OK
  9. G/A Monitor Function
     01) RUN ....................................................OK
     02) VCCDOWN ................................................OK
     03) NOCLK ..................................................OK
     04) TIMEOUT ................................................OK
 10. G/A Parallel RAM Monitor
     01) PRAM Clear .............................................OK
     02) PRAM Monitor ...........................................OK
 11. G/A Trace Function
     01) Free Trace .............................................OK
     02) Trace Stop .............................................OK
     03) Time Stamp .............................................OK
     04) RESERVED
 12. Combination
     01) B to A Time Measurement( FPGA counter ) ................OK
     02) B to A Time Measurement( G/A counter ) .................OK
     03) D to C Time Measurement( G/A counter ) .................OK
 13. Parallel Access
     01) Internal ROM Parallel Test .............................OK
     02) Internal RAM Parallel Test .............................OK
     03) Internal I/O Parallel Test .............................OK
     04) Emulation RAM Parallel Test ............................OK

Normal stopped at WWW MMM DD hh:mm:ss YYYY

Tests run for xh:xmin:xs

Summary:
    Tests performed 1 time(s).
  1. Main Board Access                    : 0 Error(s)
  2. Emulation Board Access               : 0 Error(s)
  3. Evaluation Board Access              : 0 Error(s)
  4. Basic Function                       : 0 Error(s)
  5. GO to BREAK Time Measurement         : 0 Error(s)
  6. Emulation Monitor                    : 0 Error(s)
  7. G/A Break Function                   : 0 Error(s)
  8. G/A Performance Analysis Function    : 0 Error(s)
  9. G/A Monitor Function                 : 0 Error(s)
 10. G/A Parallel RAM Monitor             : 0 Error(s)
 11. G/A Trace Function                   : 0 Error(s)
 12. Combination                          : 0 Error(s)
 13. Parallel Access                      : 0 Error(s)
```

Shows the time when diagnostic program execution ended.

Shows the execution time of the diagnostic program.

Shows the total number of errors for each test item.

H8SX/1527 and H8SX/1527R Hardware Part

RENESAS

Section 1 Overview

1.1 Notes on Usage

CAUTION

READ the following warnings before using the emulator product. Incorrect operation will damage the user system and the emulator product. The USER PROGRAM will be LOST.

1. Check all components with the component list after unpacking the emulator.
2. Never place heavy objects on the casing.
3. Observe the following conditions in the area where the emulator is to be used:
   - Make sure that the internal cooling fans on the sides of the emulator must be at least 20 cm (8") away from walls or other equipment.
   - Keep out of direct sunlight or heat. Refer to section 3.1, Environmental Conditions.
   - Use in an environment with constant temperature and humidity.
   - Protect the emulator from dust.
   - Avoid subjecting the emulator to excessive vibration. Refer to section 3.1, Environmental Conditions.
4. Protect the emulator from excessive impacts and stresses.
5. Before using the emulator's power supply, check its specifications such as power voltage and frequency.
6. When moving the emulator, take care not to subject it to strong vibration or mechanical shock.
7. After connecting the cable, check that it is connected correctly. For details, refer to section 2, Preparation before Use.
8. Supply power to the emulator and connected parts after connecting all cables. Cables must not be connected or removed while the power is on.
9. For details on differences between the target MCU and the emulator, refer to section 3.4, Support of the Target MCU.

1.2 Emulator Hardware Components

The emulator consists of an E6000H station and the E6000H's front-end unit. By installing a user system interface cable (option) to your host computer, the emulator can be connected in the same package as the device. PC interface (option) includes a PC interface board (PCI bus and PC card bus), a LAN adapter (connected with the network), and a USB adapter (connected with the USB interface). By connecting the emulator to the host computer via those interfaces, the High-performance Embedded Workshop can be used for debugging. For details on the PC interface boards (available for PCI bus and PC card bus specifications), the LAN adapter, and the USB adapter, refer to the relevant descriptive documents.

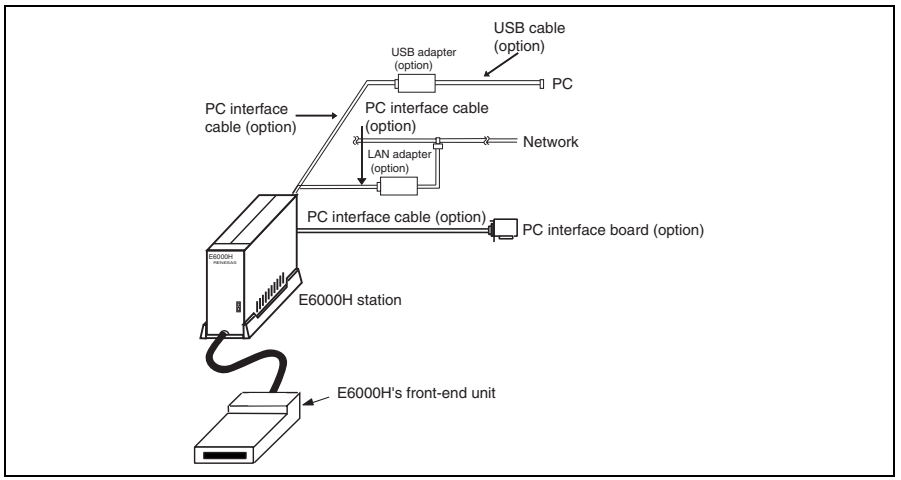


Figure 1.1 Emulator Hardware Components

1.2.1 E6000H Station Components (A Part of Photos may be Different from Real Appearances)

The names of the components on the front/rear panel of the E6000H station are listed below.

Front Panel:

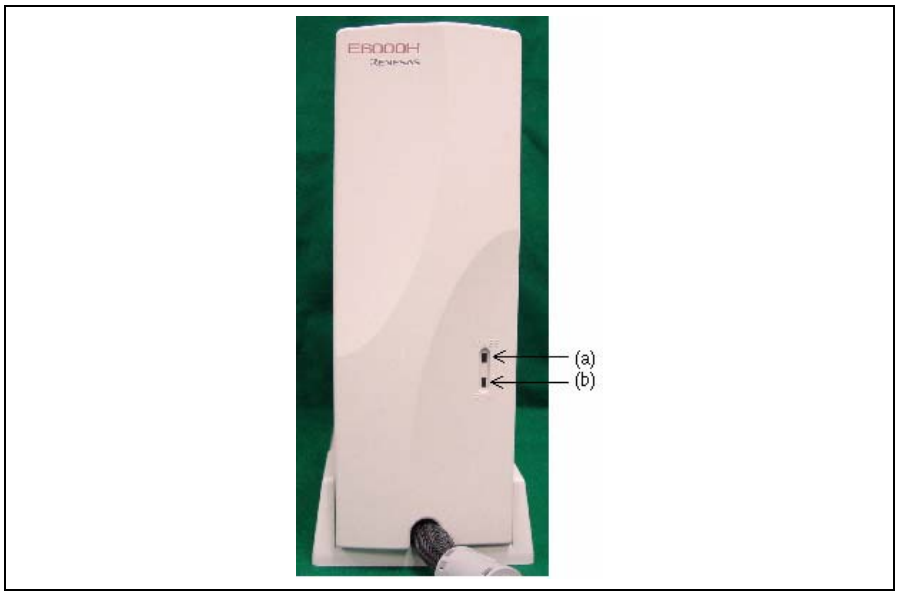


Figure 1.2 E6000H Station: Front Panel

(a) POWER lamp: Is lit up while the E6000H station is supplied with power.
(b) RUN lamp: Is lit up while the user program is running.

Rear Panel:

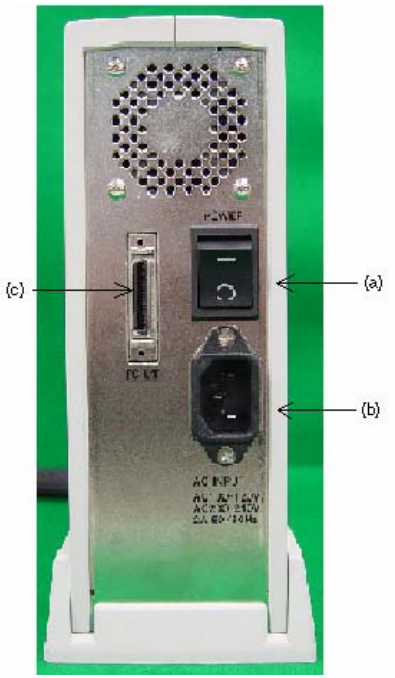


Figure 1.3 E6000H Station: Rear Panel

(a)	Power switch:	Turning this switch to I (input) supplies power to the emulator (E6000H station and the E6000H's front-end unit).
(b)	AC power connector:	For an AC 100-V to 240-V power supply.
(c)	PC interface cable connector:	For the PC interface cable that connects the host computer to the E6000H station. A PC interface board, PC card interface, LAN adapter, or USB adapter can be connected. Marked PC/IF.

1.2.2 Front-end Unit Configuration

The names of the components on the front-end unit are listed below.

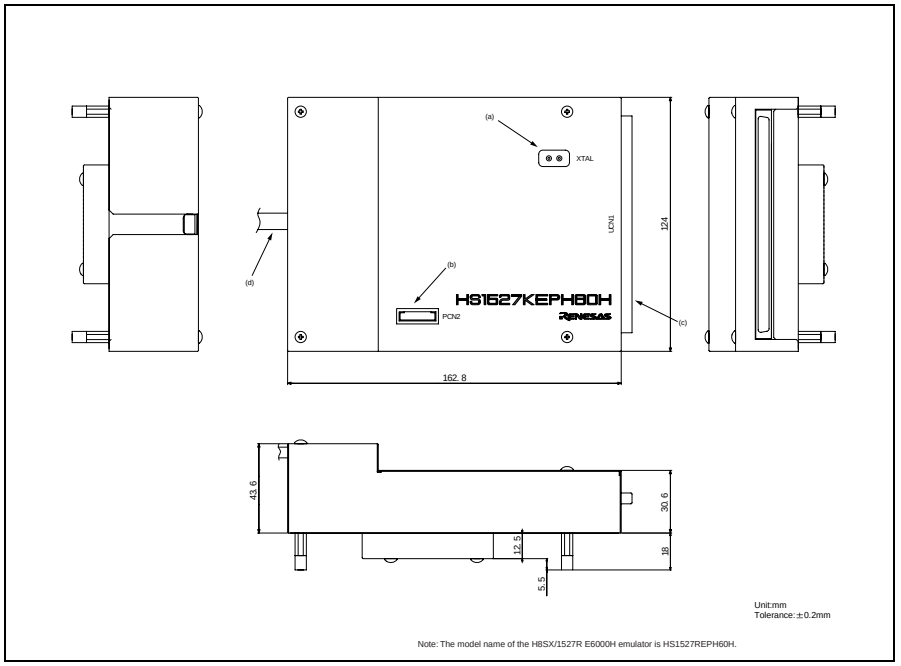


Figure 1.4 Configuration and Dimensions of the Emulator's Front-end Unit

(a) Crystal oscillator terminals:	For installing a crystal oscillator to be used as an external clock source for the MCU.
(b) External probe connector:	For connecting the external probe for external signal tracing and multibreak detection.
(c) User system interface cable connector:	For connecting the user system interface cable.
(d) Trace cable:	For connecting the front-end unit to the emulator station.

1.3 System Configuration

The emulator must be connected to a host computer via the selected PC interface board (PCI bus or PC card bus), LAN adapter, or USB adapter. For details on the PC interface boards (available for PCI bus and PC card bus specifications), the LAN adapter, and the USB adapter, refer to the relevant descriptive documents.

1.3.1 System Configuration Using Various Interfaces

(1) PC Interface Board

Figure 1.5 shows the configuration of a system in which the PC interface board is used. The emulator can be connected to a host computer via a PC interface board (option: PCI bus or PC card bus). Install the PC interface board to the expansion slot for the interface board in the host computer, and connect the interface cable supplied with the PC interface board to the emulator.

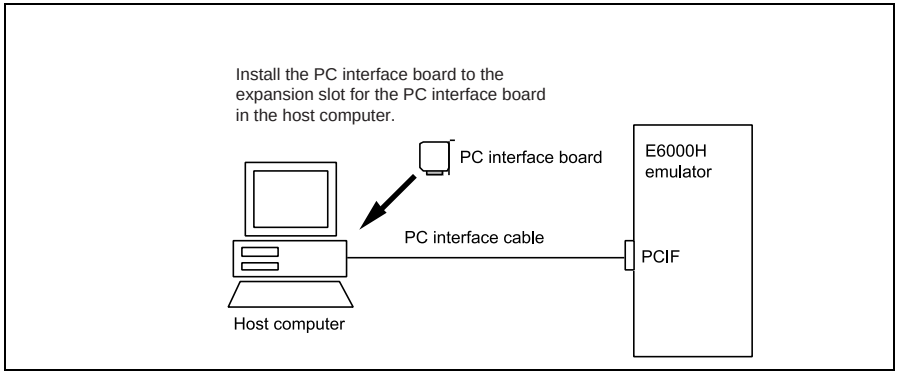


Figure 1.5 System Configuration Using a PC Interface Board

(2) LAN Adapter

Figure 1.6 shows the configuration of a system in which the LAN adapter is used. A LAN adapter can be used to connect the emulator to a host computer as a network.

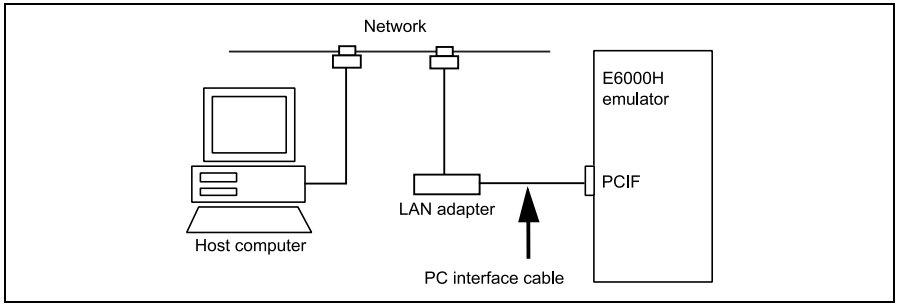


Figure 1.6 System Configuration Using a LAN Adapter

(3) USB Adapter

Figure 1.7 shows the configuration of a system in which the USB adapter is used. A USB adapter can be used to connect the emulator to a host computer with the USB interface.

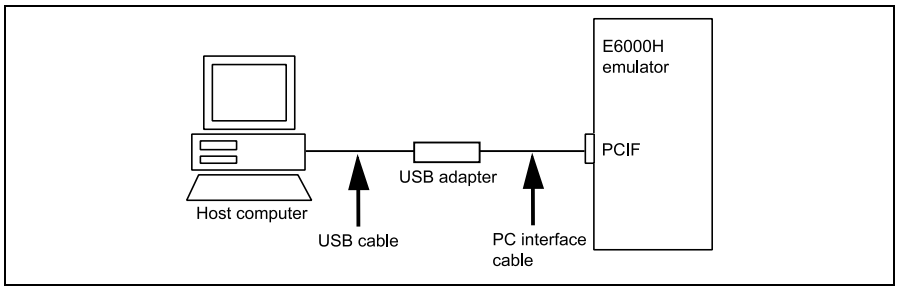


Figure 1.7 System Configuration Using a USB Adapter

Section 2 Preparation before Use

2.1 Description on Emulator Usage

This section describes the preparation before emulator usage. Figure 2.1 is a flowchart on preparation before use of the emulator.

CAUTION

Read this section and understand its contents before preparation.
Incorrect operation will damage the user system and the emulator.
The USER PROGRAM will be LOST.

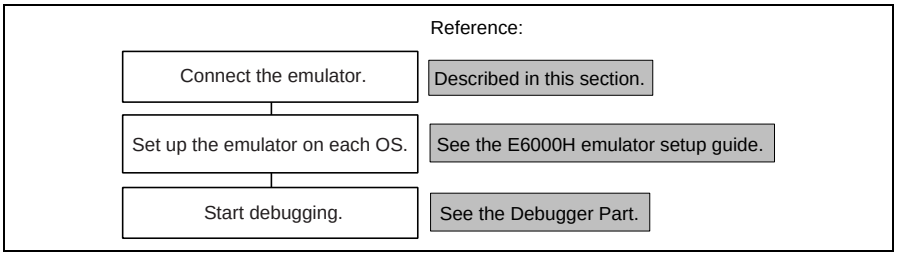


Figure 2.1 Emulator Preparation Flowchart

2.2 Emulator Connection

The following description covers connection of the emulator.

2.2.1 Connecting the Emulator to the User System

⚠WARNING

Always switch OFF the emulator and user system before connecting or disconnecting any CABLES. Failure to do so will result in a FIRE HAZARD and will damage the user system and the emulator or will result in PERSONAL INJURY. The USER PROGRAM will be LOST.

1. Check that the emulator power switch is turned off. Ensure that the power lamp on the right side of the E6000H station's front panel is not lit.
2. Remove the AC power cable of the E6000H station from the outlet (if the cable is connected to the outlet).
3. The emulator is connected to the user system via the user system interface cable.

2.2.2 Connecting the User System Interface Cable

⚠WARNING

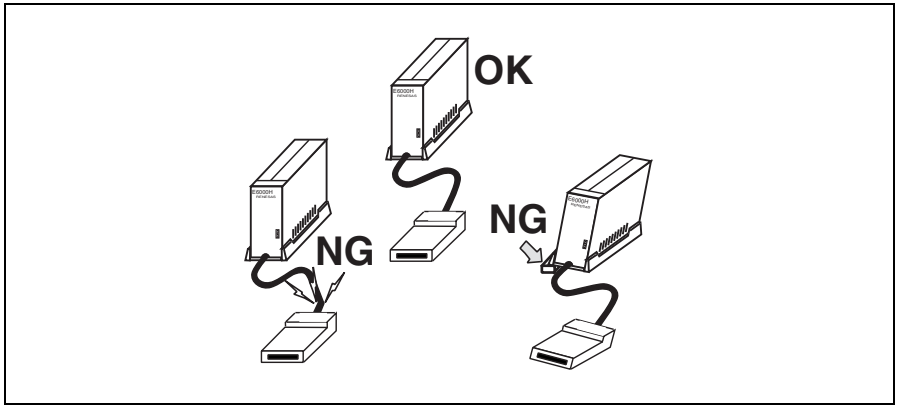
Always switch OFF the emulator and user system and check pin numbers on the connectors and IC socket before connecting or disconnecting the USER SYSTEM INTERFACE CABLE. Connection with the power on or incorrect connection will damage the emulator, user system interface cable, and user system, and result in a FIRE HAZARD.

For details on the method of connecting the user system interface cable, refer to the descriptions of the user system interface cables for individual H8SX/1527 E6000H-series products.

2.2.3 Precautions on Connecting the User System

When connecting the emulator to the user system, note the following:

1. Secure the E6000H station location.
   Place the E6000H station and the E6000H's front-end unit so that the trace cable is not bent or twisted, as shown below. A bent or twisted cable will impose stress on the user interface, leading to connection or contact failure. Make sure that the E6000H station and the front-end unit are placed in a secure position so that they do not move and impose stress on the user interface while being used.



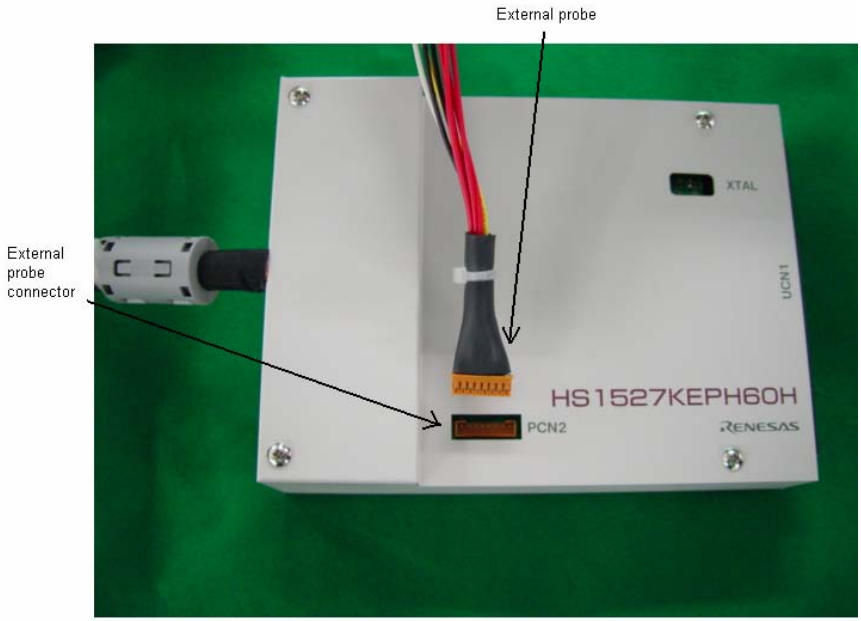
2. Make sure the power supply is off.
   Before connecting the emulator to the user system, check that the emulator and the user system are turned off.
3. Connect Vcc to the user system power.
   The emulator monitors and determines whether the user system is turned on or off by the Vcc pins.
   After connecting the user system to the emulator via the user system interface cable, be sure to supply power to the Vcc pins. Otherwise, the emulator assumes that the user system is not connected.
   When the user system is connected, check that the power of the user system is supplied to these pins.

2.2.4 Connecting the External Probe

CAUTION

Check the external probe direction and connect the external probe to the emulator station correctly. Incorrect connection will damage the probe or connector.

When an external probe is connected to the external probe connector on the emulator's front-end unit, it enables external signal tracing and multibreak detection. Figure 2.2 shows the external probe connector.



Note: The model name of the H8SX/1527R E6000H emulator is HS1527REPH60H.

Pin No.	Probe Name	Signal Name	Note
1	1	Probe input 0	Synchronous break input pin
2	2	Probe input 1	
3	3	Probe input 2	
4	4	Probe input 3	
5	G	GND	GND connection pin
6	TUV	Trigger output	Trigger mode output pin
7	G	GND	GND connection pin
8	T5V	RUN or break output	High-level output at RUNSTEP

Figure 2.2 External Probe Connector

2.2.5 Selecting the Clock

This emulator supports three types of clock for the MCU: a crystal oscillator attached on the emulator's front-end unit, external clock input from the user system, and the emulator internal clock. The clock is specified with the [Configuration] dialog box.

This emulator can use a clock source (ϕ) running at up to 40.0 MHz as the H8SX/1527 clock input. The H8SX/1582 or H8SX/1527R can operate with clock sources (ϕ) running at up to 48.0 MHz.

CLOCK command —— Xtal (Crystal oscillator: 4 to 9 MHz)

Target (External clock: 4 to 9 MHz)

5 (Emulator internal clock: 5 MHz)

6 (Emulator internal clock: 6 MHz)

Crystal Oscillator: A crystal oscillator is not supplied with the emulator. Prepare and use one that has the same frequency as that of the user system. When using a crystal oscillator as the MCU clock source, the frequency range must be from 4 to 9 MHz.

CAUTION

Always switch OFF the emulator and user system before connecting or disconnecting the CRYSTAL OSCILLATOR. Otherwise, the USER PROGRAM will be LOST.

Follow the procedure listed below to install the crystal oscillator:

1. Check that the emulator power switch is turned off. (Check that the power LED is not lit.)
2. Attach the crystal oscillator into the terminals on the emulator's front-end unit (figure 2.3). Be careful to ensure that the terminals do not touch the casing.
3. Turn on the user system power and then the emulator power. The crystal oscillator will then be automatically set and started up. This function will allow the execution of the user program at the operating frequency of the user system even when the user system is not connected to the emulator.

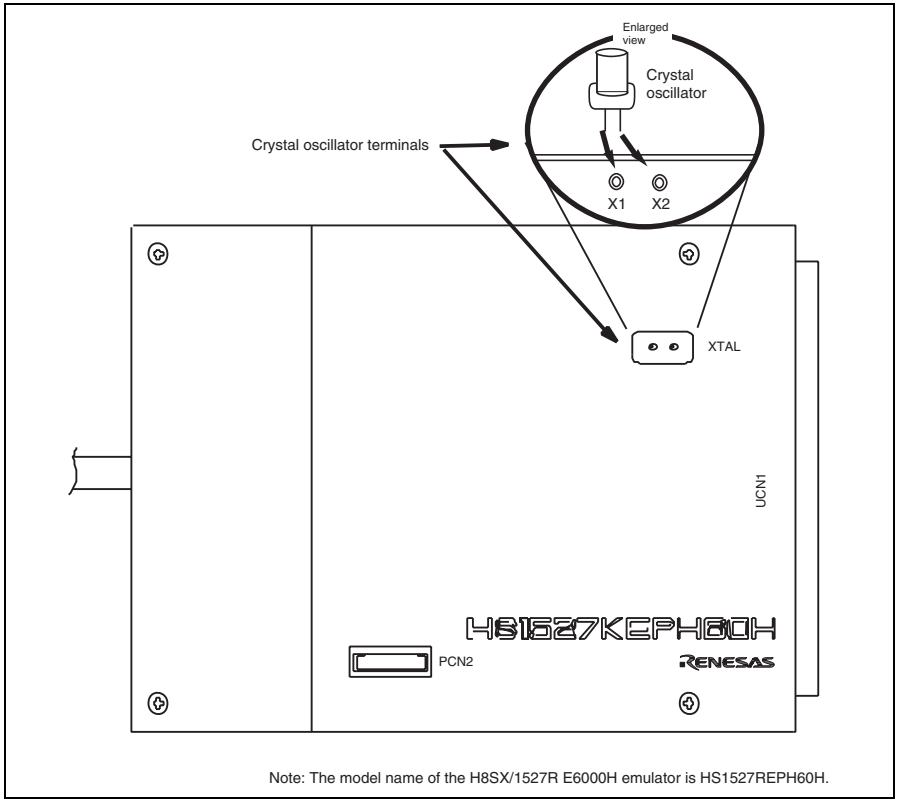


Figure 2.3 Installing the Crystal Oscillator

External Clock: Follow the procedure listed below to select the external clock.

1. Check that the emulator power switch is turned off. (Check that the power LED is not lit.)
2. Connect the user system interface cable to the user system and supply a clock signal through the EXTAL pin from the user system.
3. Turn on the user system power and then the emulator power. The external clock source will then be automatically specified.

Emulator Internal Clock: Specify 5.0 MHz or 6.0 MHz in the [Configuration] dialog box.

Reference:

When the emulator system program is initiated, the emulator automatically selects the MCU clock source according to the following priority:

1. User system's clock when an external clock is supplied from the user system
2. Crystal oscillator, if one is mounted on the emulator's front-end unit
3. Emulator-internal clock

2.2.6 Connecting the System Ground

CAUTION

Separate the frame ground from the signal ground at the user system. When the frame ground is connected to the signal ground and the emulator is then connected to the user system, the emulator will malfunction.

The emulator's signal ground is connected to the user system's signal ground via the emulator's front-end unit. In the E6000H station, the signal ground and frame ground are connected (figure 2.4). At the user system, connect the frame ground only; do not connect the signal ground to the frame ground.

If it is difficult to separate the frame ground from the signal ground in the user system, ground the frame to the same outlet as the 100-V to 240-V AC power supply of the emulator station (figure 2.5) so that the ground potentials become even.

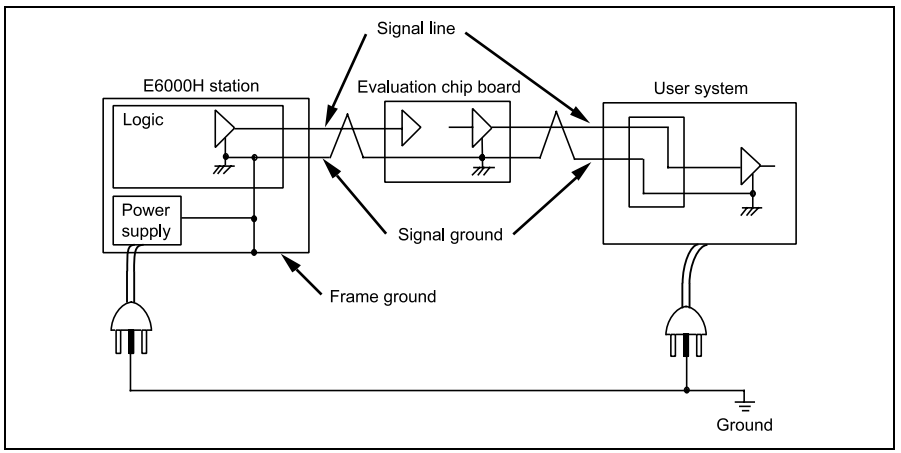


Figure 2.4 Connecting the System Ground

WARNING

Always switch OFF the emulator and user system before connecting or disconnecting any CABLES. Failure to do so will result in a FIRE HAZARD and will damage the user system and the emulator or will result in PERSONAL INJURY. The USER PROGRAM will be LOST.

The user system must be connected to an appropriate ground so as to minimize noise and the adverse effects of ground loops. When connecting the emulator's front-end unit and the user system, confirm that the ground pins of the front-end unit are firmly connected to the user system's ground.

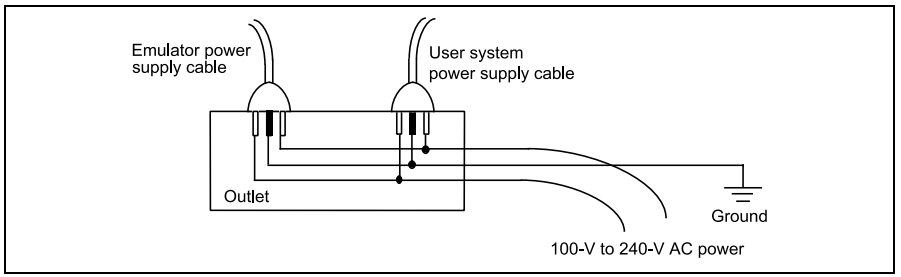


Figure 2.5 Connecting the Frame Ground

2.2.7 PC Interface Board Specifications

For details on the PC interface board, LAN adapter, or USB adapter, refer to their description notes.

Section 3 Hardware Specifications

3.1 Environmental Conditions

CAUTION

Observe the conditions listed in table 3.1 when using the emulator. The following environmental conditions must be satisfied, otherwise the user system and the emulator will not operate normally. The USER PROGRAM will be LOST.

Table 3.1 Environmental Conditions

Item	Specifications	
Temperature	Operating:	+10 to +35°C
	Storage:	−10 to +50°C
Humidity	Operating:	35 to 80% RH, no condensation
	Storage:	35 to 80% RH, no condensation
Vibration	Operating:	2.45 m/s^2 max.
	Storage:	4.9 m/s^2 max.
	Transportation:	14.7 m/s^2 max.
AC input power	Voltage:	100 V to 240 V AC
	Frequency:	50/60 Hz
	Power consumption:	75 W
Ambient gases	There must be no corrosive gases present.	

3.2 Emulator External Dimensions and Mass

Figure 3.1 shows the external dimensions and mass of the E6000H emulator.

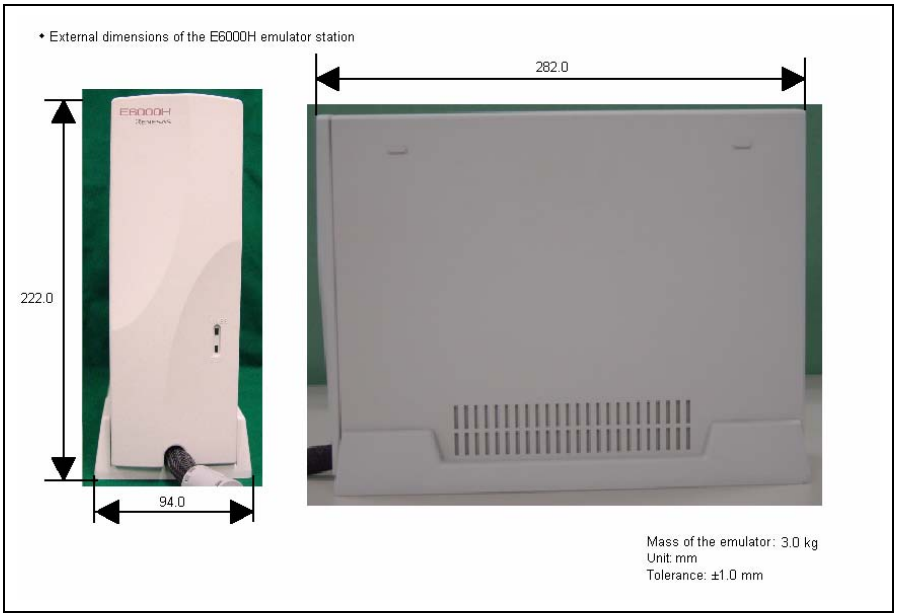


Figure 3.1 External Dimensions and Mass of the Emulator

3.3 User System Interface Circuit

3.3.1 User System Interface Circuit

The circuits that interface the MCU in the emulator to the user system include the level-shifter circuits (for 5 V and 3.3 V) and resistors. When connecting the emulator to a user system, adjust the user system hardware compensating for FANIN, FANOUT, and propagation delays.

The user system interface circuits connected to the user system are shown in figure 3.2.

The delay time is generated on the timing of the _RES and _NMI signals when they are input to the MCU from the user system, as shown in table 3.2, because this connection for those signals is via logic circuit on the emulator's front-end unit.

Table 3.2 Delay Time for Signal Connected via the Emulator's Front-end Unit

Signal Name	Delay Time (ns)
_RES	21
_NMI	21

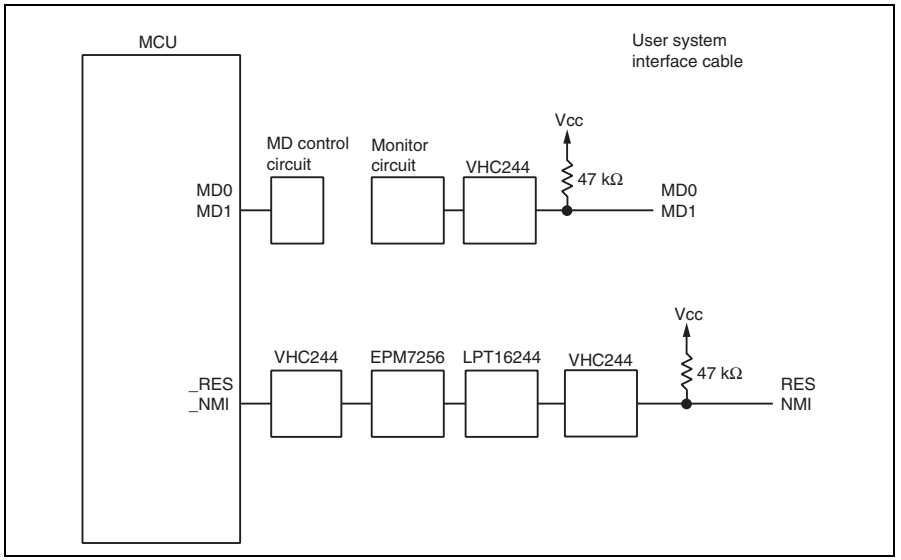


Figure 3.2 User System Interface Circuits (1)

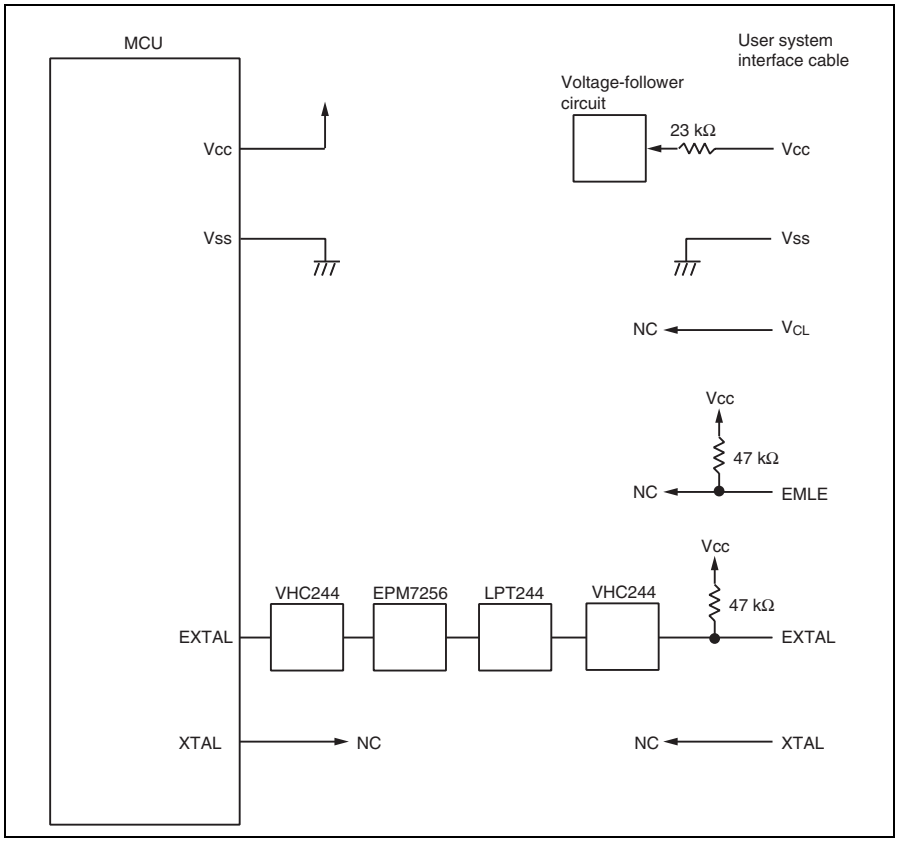


Figure 3.2 User System Interface Circuits (2)

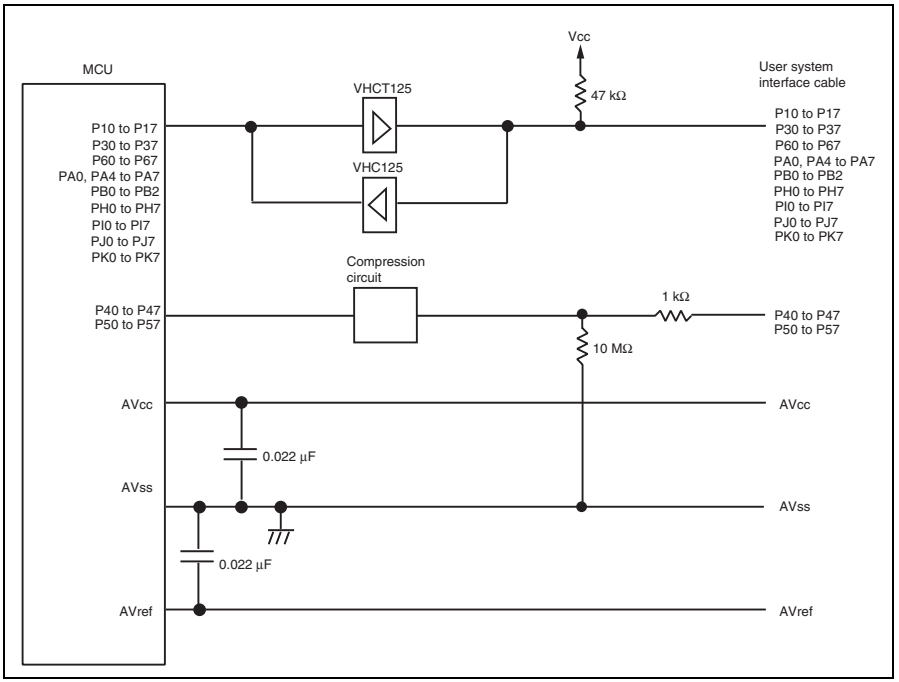


Figure 3.2 User System Interface Circuits (3)

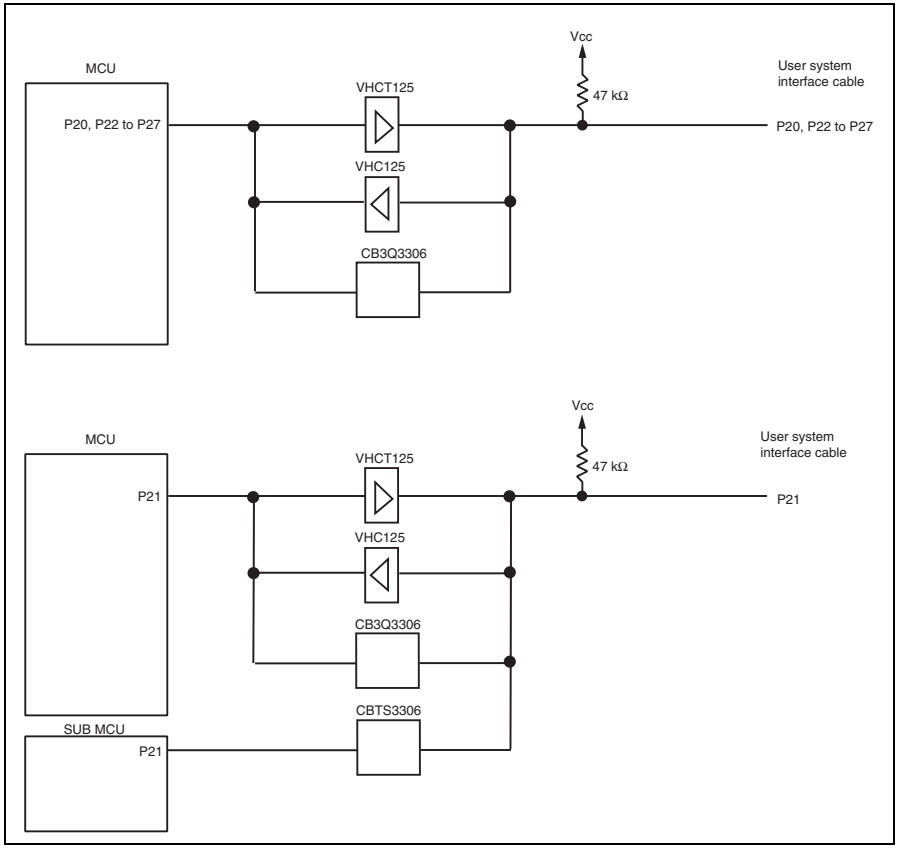


Figure 3.2 User System Interface Circuits (4)

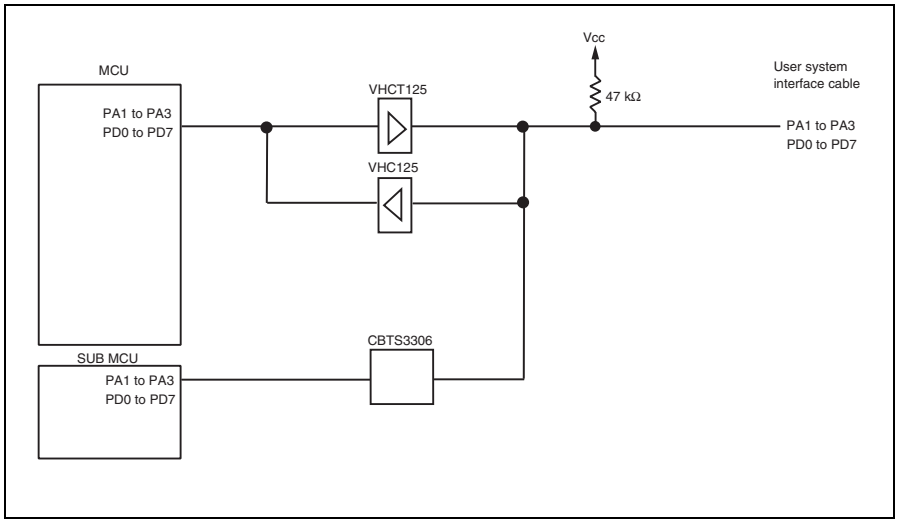


Figure 3.2 User System Interface Circuits (5)

3.4 Support of the Target MCU

3.4.1 Memory Space

The architecture of the MCU allows for a 16-Mbyte memory space.

(1) On-Chip I/O Area

If an attempt is made to access the on-chip I/O area, the on-chip I/O area in the MCU installed in the emulator is accessed. To break user program execution when the on-chip I/O area is written to or accessed, use the hardware break or internal break.

(2) On-Chip ROM/RAM Area

If an attempt is made to access the on-chip ROM/RAM area, the on-chip ROM/RAM area in the MCU installed in the emulator is accessed. To break user program execution when the on-chip ROM/RAM area is written to or accessed, use the hardware break or internal break.

3.4.2 Power-Down Modes (Sleep and Software Standby)

For reduced power consumption, the MCU has sleep and software standby modes.

(1) Sleep and Software Standby Modes

- Break

  The MCU can be taken out of the sleep and software standby modes either in the normal ways or through satisfaction of a break condition (forced break). When restarting after a break, the user program will restart at the instruction following the SLEEP instruction.
- Trace

  Trace information is not acquired in these modes.
- Memory access with emulator functions

  For information on displaying and modifying the contents of memory in the sleep and software standby modes, refer to section 5.4, Displaying and Modifying the Contents of Memory, in the Debugger Part.

3.4.3 Interrupts

During execution and step execution, the user can interrupt the MCU.

Interrupt sources are retained while emulation is halted (break mode). In such cases, interrupt processing commences immediately after emulation is restarted.

3.4.4 Control Input Signals (/RES and /NMI)

The MCU control input signals are /RES and /NMI. The /RES and /NMI signals are only valid when emulation was started with normal program execution (i.e., they are invalid when emulation was started with step execution).

The input of the /RES or /NMI signal during execution or step execution can be disabled by a setting in the [Configuration] dialog box.

3.4.5 Watchdog Timer (WDT)

When emulation is suspended (i.e. by a break), counting up by the WDT timer counter (TCNT) is also suspended. Counting resumes when emulation is resumed (user mode).

During break mode, a prescaler, which supplies a clock to TCNT, operates continuously. Since the prescaler might be in different phases before and after emulation goes through a period in the break mode, a break can change the period before the WDT overflows by ±1 cycle of the prescaler's clock.

3.4.6 A/D Converter

As well as analog input pins, the A/D converter has AVcc, AVss, and /ADTRG pins. As the A/D converter operates with an independent power supply, connect AVcc (the power supply pin) to the A/D power supply on the user system.

Notes:
1. When the A/D converter is not in use, connect AVcc to Vcc.
2. As there are user system interface cable, compression circuits (for compression at the ratio of 5:3.3), wiring on the printed circuit board, and protective circuits are connected between the user system and the MCU on the evaluation chip board, the precision of conversion is lower than that of the actual MCU.

3.4.7 Emulator State and Internal Modules

Operation of some internal modules depends on the emulator's state. Tables 3.3 and 3.4 show the relation between the emulator's state and operation of the internal modules.

Table 3.3 State of the H8SX/1527 Emulator and Operation of Internal Modules

Internal Module	Operation while Emulation is Halted (Break)	Operation During Emulation (Execution or Step Execution)
DMAC (DMA controller)	Yes	Yes
TPU (16-bit timer pulse unit)	Yes	Yes
PPG (programmable pulse generator)	Yes	Yes
WDT (watchdog timer)	No	Yes
SCI (serial communication interface)	Yes	Yes
A/D converter	Yes	Yes
HCAN (controller area network)	Yes	Yes
SSU (synchronous serial communication unit)	Yes	Yes
I/O port	Yes	Yes
H-UDI (user debugging interface)	Not available*	Not available*

Note: The user cannot use the H-UDI because it is being used by the emulator.

Table 3.4 State of the H8SX/1527R Emulator and Operation of Internal Modules

Internal Module	Operation while Emulation is Halted (Break)	Operation During Emulation (Execution or Step Execution)
DMAC (DMA controller)	Yes	Yes
TPU (16-bit timer pulse unit)	Yes	Yes
PPG (programmable pulse generator)	Yes	Yes
WDT (watchdog timer)	No	Yes
SCI (serial communication interface)	Yes	Yes
A/D converter	Yes	Yes
RCAN (controller area network)	Yes	Yes
SSU (synchronous serial communication unit)	Yes	Yes
I/O port	Yes	Yes
H-UDI (user debugging interface)	Not available*	Not available*

Note: The user cannot use the H-UDI because it is being used by the emulator.

3.4.8 Differences in Values of Registers

Note that certain general and control registers of both emulators are initialized whenever the system is activated or the MCU is reset by a command.

Table 3.5 Initial Values of Registers in the MCU and the Emulator

	Emulator		
Register Name	**Power On**	**Reset (Reset CPU)**	**MCU (Power-On Reset)**
PC	PC value indicated by the power-on reset vector	PC value indicated by the power-on reset vector	Undefined
ER0 to ER7	H'00000000	Value before the reset	Undefined
CCR	B'1XXXXXXX	B'1XXXXXXX	B'1XXXXXXX
EXR	B'01111111	Value before the reset	B'01111111
MACH	H'00000000	Value before the reset	Undefined
MACL	H'00000000	Value before the reset	Undefined
VBR	H'00000000	Value before the reset	H'00000000
SBR	H'FFFFFF00	Value before the reset	H'FFFFFF00

Note: X indicates an undefined bit.

3.4.9 Differences in Access to the Internal RAM

In the H8SX/1527 E6000H and H8SX/1527R E6000H emulators, writing to the internal RAM takes one cycle by default. To make the emulator write data to the RAM in two cycles, the user program must write the value H'35 to address H'FFFFFD97 before any RAM access. After that, the RAM should not be accessed for at least one cycle. This address will be reinitialized whenever the reset pin is activated (goes low). In such cases, write the value H'35 to the same address again after the system has been released from the reset state.

In the actual MCU (H8SX/1527 or H8SX/1527R), writing to the RAM takes a fixed two cycles and H'35 is always read from address H'FFFFFD97. Writing to the address is invalid.

3.5 Notes Specific to the H8SX/1527 E6000H and H8SX/1527R E6000H Emulators

3.5.1 Custom Device Function

The maximum value selectable for the on-chip ROM with the custom device functions of the H8SX/1527 E6000H and H8SX/1527R E6000H emulators is 2 Mbytes. Do not specify a value greater than 2 Mbytes.

3.5.2 Non-availability of P2 Open-Drain Outputs

In the emulator, port P2 pins cannot be set up as NMOS open-drain outputs by settings in register P2ODR. Do not set any bit in P2ODR to 1.

3.5.3 Limitations on Control of the SSU Pins

(1) Switching between Operation as an SSCK Pin and an I/O Port Pin after Conflict Errors

If a conflict error occurs while the values of bits MSS and SCKS in an SSCRH register are 1, the MSS bit will be cleared and the corresponding SSCK pin will be switched to act as an input. After that, even if the user sets the SCKS bit to 0 to reconfigure the SSCK pin to act as an I/O port pin, the emulator will not set the SSCK pin up as an I/O port pin. To allow use of the SSCK pin as an I/O port pin, have the program write 0 to the MSS bit if a conflict error occurs. This procedure is not necessary unless the SSCK pin is to be used as an I/O port pin.

(2) Operation of Port Pins that can also Function as SSU Pins (Ports 2, A, and D)

(a) When the SSU pins are set up as inputs

In the actual MCU, the state of an SSU pin operating as an input can be read from the corresponding port register. Note, however, that the value read is undefined in the case of the emulator.

(b) When the SSU pins are set up as outputs

In the actual MCU, the state of an SSU pin operating as an output can be read from the corresponding port register as long as the value of the corresponding data direction register bit is 0. Note, however, that the value read is undefined in the case of the emulator.

(3) Input Buffer Control Register (PnICR)

In the emulator, pins can be used as inputs for the peripheral modules listed below regardless of the setting of the corresponding ICR bits.

In the actual MCU, however, the input buffers are invalid and the input signals are fixed high for pins where the corresponding ICR bit has been cleared to 0. A pin cannot be used as an input for a peripheral module unless the corresponding ICR bit is set to 1.

If you are using any of these modules, ensure that your program sets the ICR bits for the corresponding pins to 1 before the module is used.

Peripheral modules in the H8SX/1527 E6000H: SSU, HCAN, and RTIP

Peripheral modules in the H8SX/1527R E6000H: SSU, RCAN, and RTIP

(4) Restriction on the SSU Module Stop Function

If the SSU module-stop function is used, the emulator can independently change the module stop bit for each channel to take it out of the module-stopped state and make the channel accessible. Setting the module-stop bit for an SSU channel to 1 independently makes that SSU channel inaccessible. To initialize the SSU channels, however, note that all three module-stop bits, i.e. MSTPC8, MSTPC9, and MSTPC10, must be set to 1. For details on the SSU module-stop function of the MCU, refer to the hardware manual.

Section 4 Diagnostic Test Procedure

This section describes the diagnostic test procedure using the E6000H diagnostic program.

4.1 System Set-Up for Diagnostic Program Execution

To execute the diagnostic program, use the following hardware; do not connect the user system interface board and user system.

- E6000H (HS1527KEPH60H or HS1527REPH60H)
- Host computer
- The E6000 PC interface board which will be one of the following boards:
  PCI bus interface board (HS6000EIC01H or HS6000EIC02H)
  PC card interface (HS6000EIP01H)
  LAN adapter (HS6000ELN01H)
  USB adapter (HS6000EIU01H or HS6000EIU02H)

1. Install the E6000 PC interface board in the host computer and connect the supplied PC interface cable to the board.
2. Connect the PC interface cable to the emulator.
3. Connect the supplied AC power cable to the emulator.
4. Boot the host computer up and enter the command input wait state of DOS (the DOS prompt) (Windows® 98SE or Windows® Me) or the command prompt (Windows NT®, Windows® 2000, or Windows® XP). If the prompt window is not set up to be displayed over the whole screen, press the [Alt + Enter] key chord to switch the mode. To return the prompt to a window, press the [Alt + Enter] key. This key chord works in the same way regardless of the OS being used.

Note: If the display of the MS-DOS prompt does not switch from full-screen to windowed or vice-versa in response to the [Alt + Enter] key chord, mark the [Alt+Enter] check box in [Windows shortcut keys] of the [Misc] page on the [MS-DOS Prompt Properties] dialog box and click the [Apply] button to update the settings.

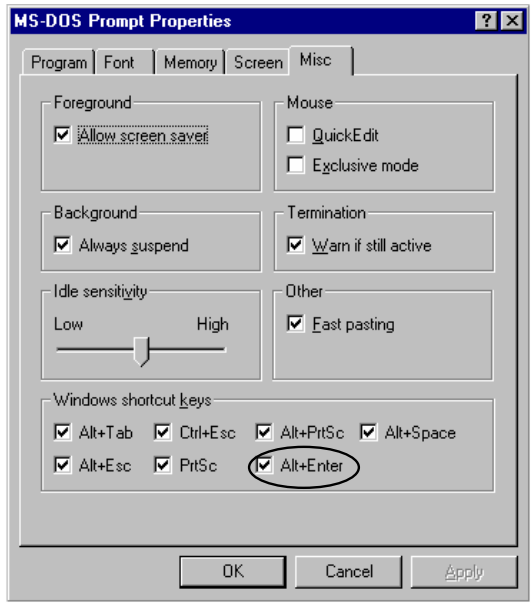


Figure 4.1 [Misc] Page

5. Turn on the E6000H emulator switch.

Note: To execute the diagnostic program, firstly turn on the power of the emulator. The diagnostic program checks the initial state of the hardware. Therefore, after turning on the power, do not activate the High-performance Embedded Workshop before executing the diagnostic program.

4.2 Test Item of the Diagnostic Program (HS1527KEPH60H)

Table 4.1 shows the test items of this diagnostic program.

Table 4.1 Test Items of the Diagnostic Program

Test No.	Test Item	Description
1	Main Board Access	Register test in the E6000H main board
2	Emulation Board Access	Register test in the E6000H emulation board
3	Evaluation Board Access	Register test in the E6000H's front-end unit
4	Basic Function	Test of the basic functions
5	GO to BREAK Time Measurement	Test of the execution time measurement function
6	Emulation Monitor	Test of the emulation monitor
7	G/A Break Function	Test of the G/A break function
8	G/A Performance Analysis Function	Test of the G/A performance measurement function
9	G/A Monitor Function	Test of the G/A monitoring function
10	G/A Parallel RAM Monitor	Test of the G/A parallel RAM monitoring function
11	G/A Trace Function	Test of the G/A trace function
12	Combination	Test of several functions in combination
13	Parallel Access	Test of the parallel access function
14	H8SX/1527 Register Read/Write	Test of registers in the SUB MCU
15	FPGA Parallel RAM Function	Test of the FPGA's parallel RAM monitoring function

4.3 Diagnostic Test Procedure Using the Diagnostic Program

Insert the CD-R (HS1650EPH60SR supplied with the emulator) into the CD-ROM drive of the host computer, use the command prompt to change the current directory to <Drive>:\Diag\1527, and enter the command below that corresponds to the PC interface board used to initiate the diagnostic program:

1. PCI bus interface board (HS6000EIC01H or HS6000EIC02H)
   > TM1527 –PCI (Enter)

2. PC card interface (HS6000EIP01H)
   > TM1527 –PCCD (Enter)

3. LAN adapter (HS6000ELN01H)
   > TM1527 –ELN (Enter)

4. USB adapter (HS6000EIU01H or HS6000EIU02H)
   > TM1527 –USB (Enter)

The High-performance Embedded Workshop must be installed before the test program is executed.

Be sure to initiate the diagnostic program from <Drive>:\Diag\1527. Do not initiate it from a directory other than <Drive>:\Diag\1527, such as > <Drive>:\Diag\1527\TM1527 –PCI (Enter). If the diagnostic program is initiated when the current directory is not <Drive>:\Diag\1527, the diagnostic program will not operate correctly.

When –S is added to the command line, as in > TM1527 –PCI –S (Enter), steps 1 to 15 will be repeatedly executed. To stop execution in this case, enter Q.

Notes: 1. <Drive> is the drive name of the CD-ROM drive.
2. Do not remove the CD-R from the CD-ROM drive during execution of the test program.

The following messages are displayed during the test. There are 15 steps in this test.

Message	Description
`H8SX/1527 E6000H Emulator Tests V*.*` `Copyright (c) 2003 Renesas Technology Corp.`	Test program start message. x.x shows the version number.
`  Loading driver .....OK (Use PCI)`	Shows that the PC interface board is correctly installed in the host computer.
`  Initializing driver .....OK`	
`  Searching for interface card .....OK`	
`  Checking emulator is connected .....OK`	Shows that the E6000H emulator is correctly connected to the host computer.
`Emulator board information:` `    Main board ID:   H'*    Emulation board ID:  H'***`	Shows the ID number of the E6000H emulator.
`Normal started at WWW MMM DD hh:mm:ss YYYY`	Shows the time when the diagnostic program was started up.

```
******* NORMAL TEST - Press 'Q' to stop *******          (COUNT=0001)
  1. Main Board Access
     01) Registers Initial Value Check .....OK
     02) Registers Write/Verify .....OK
     03) DPRAM Address Decode Test .....SKIP
     04) DPRAM Marching Test .....SKIP
     05) Trace Memory Address Decode Test .....OK
     06) Trace Memory Marching Test .....OK
     07) G/A Registers Initial Value Check .....OK
     08) G/A Registers Write/Verify .....OK
  2. Emulation Board Access
     01) Registers Initial Value Check .....OK
     02) Registers Write/Verify .....OK
  3. Evaluation Board Access
     01) Registers Initial Value Check .....OK
     02) Registers Write/Verify .....OK
     03) H-UDI IDCODE Check .....OK
     04) Firmware BOOT .....OK
     05) Configuration Set .....OK
  4. Basic Function
     01) GO to BREAK .....OK
     02) RESET GO .....OK
     03) STEP .....OK
     04) KEYBREAK .....OK
     05) BRKCONT .....OK
     06) Internal ROM Address Decode Test .....OK
     07) Internal ROM Marching Test .....OK
     08) Internal RAM Address Decode Test .....OK
     09) Internal RAM Marching Test .....OK
     10) RESERVED
     11) RESERVED
     12) MAP Break .....OK
  5. GO to BREAK Time Measurement
     01) Counter Test Mode (EMU 12MHz MPU 12MHz Sampling 20ns) ..OK
     02) EMU 12Mhz MPU 12Mhz Sampling 20ns .....OK
     03) EMU 12Mhz MPU 12Mhz Sampling 1.6us .....OK
     04) EMU 12Mhz MPU 12Mhz Sampling 52us .....OK
     05) EMU 12Mhz MPU 12Mhz Sampling MPU .....OK
     06) EMU 12Mhz MPU 12Mhz Sampling MPU/2 .....OK
     07) EMU 12Mhz MPU 12Mhz Sampling MPU/4 .....OK
     08) EMU 12Mhz MPU 12Mhz Sampling MPU/8 .....OK
     09) EMU 12Mhz MPU 48Mhz Sampling 20ns .....OK
     10) EMU  4Mhz MPU 32Mhz Sampling 20ns .....OK
```

```
     11) EMU 35.7Mhz MPU 35.7Mhz Sampling 20ns ..................OK
     12) EMU 35.9Mhz MPU 35.9Mhz Sampling 20ns ..................OK
     13) EMU f1Mhz MPU f2Mhz Sampling 20ns ......................SKIP
  6. Emulation Monitor
     01) RESERVED
     02) RESERVED
     03) ASEST ..................................................OK
  7. G/A Break Function
     01) Address Condition ......................................OK
     02) Data Condition .........................................OK
     03) Control Signal Condition (ASEDSHH/HL/LH/HL) ............OK
     04) Function Code Condition (ASEST3-0) .....................OK
     05) RESERVED
     06) Control Signal Condition (CB22-19) .....................OK
     07) Control Signal Condition (CB18-16) .....................OK
  8. G/A Performance Analysis Function
     01) Time Measurement (20ns Sampling) .......................OK
  9. G/A Monitor Function
     01) RUN ....................................................OK
     02) VCCDOWN ................................................SKIP
     03) NOCLK ..................................................OK
     04) TIMEOUT ................................................OK
 10. G/A Parallel RAM Monitor
     01) PRAM Clear .............................................OK
     02) PRAM Monitor ...........................................OK
 11. G/A Trace Function
     01) Free Trace .............................................OK
     02) Trace Stop .............................................OK
     03) Time Stamp .............................................OK
 12. Combination
     01) B to A Time Measurement( FPGA counter ) ................OK
     02) B to A Time Measurement( G/A counter ) .................OK
     03) D to C Time Measurement( G/A counter ) .................OK
 13. Parallel Access
     01) Internal ROM Parallel Test .............................OK
     02) Internal RAM Parallel Test .............................OK
     03) Internal I/O Parallel Test .............................OK
 14. H8SX/1527 Register Read/Write
     01) Register Initial Value Check ...........................OK
     02) Register Write/Verify ..................................OK
 15. FPGA Parallel RAM Function
     01) CH0 256Byte Area Check(Byte) ......................................OK
     02) CH0 256Byte Area Check(Word) ......................................OK
     03) CH0 256Byte Area Check(Long Word) .................................OK
     04) CH1-CH11 256Byte Area Check(Long Word) ............................OK

Normal stopped at WWW MMM DD hh:mm:ss YYYY

Tests run for xh:xmin:xs

Summary:
    Tests performed 1 time(s).
  1. Main Board Access                  : 0 Error(s)
  2. Emulation Board Access             : 0 Error(s)
  3. Evaluation Board Access            : 0 Error(s)
  4. Basic Function                     : 0 Error(s)
  5. GO to BREAK Time Measurement       : 0 Error(s)
  6. Emulation Monitor                  : 0 Error(s)
  7. G/A Break Function                 : 0 Error(s)
  8. G/A Performance Analysis Function  : 0 Error(s)
  9. G/A Monitor Function               : 0 Error(s)
 10. G/A Parallel RAM Monitor           : 0 Error(s)
 11. G/A Trace Function                 : 0 Error(s)
 12. Combination                        : 0 Error(s)
 13. Parallel Access                    : 0 Error(s)
 14. H8SX/1527 Register Read/Write      : 0 Error(s)
 15. FPGA Parallel RAM Function         : 0 Error(s)
```

Shows the time when diagnostic program execution ended.

Shows the execution time of the diagnostic program.

Shows the total number of errors for each test item.

4.4 Test Item of the Diagnostic Program (HS1527REPH60H)

Table 4.2 shows the test items of this diagnostic program.

Table 4.2 Test Items of the Diagnostic Program

Test No.	Test Item	Description
1	Main Board Access	Register test in the E6000H main board
2	Emulation Board Access	Register test in the E6000H emulation board
3	Evaluation Board Access	Register test in the E6000H's front-end unit
4	Basic Function	Test of the basic functions
5	GO to BREAK Time Measurement	Test of the execution time measurement function
6	Emulation Monitor	Test of the emulation monitor
7	G/A Break Function	Test of the G/A break function
8	G/A Performance Analysis Function	Test of the G/A performance measurement function
9	G/A Monitor Function	Test of the G/A monitoring function
10	G/A Parallel RAM Monitor	Test of the G/A parallel RAM monitoring function
11	G/A Trace Function	Test of the G/A trace function
12	Combination	Test of several functions in combination
13	Parallel Access	Test of the parallel access function
14	H8SX/1527R Register Read/Write	Test of registers in the SUB MCU
15	FPGA Parallel RAM Function	Test of the FPGA's parallel RAM monitoring function

4.5 Diagnostic Test Procedure Using the Diagnostic Program

Insert the CD-R (HS1650EPH60SR supplied with the emulator) into the CD-ROM drive of the host computer, use the command prompt to change the current directory to <Drive>:\Diag\1527R, and enter the command below that corresponds to the PC interface board used to initiate the diagnostic program:

1. PCI bus interface board (HS6000EIC01H or HS6000EIC02H)
   > TM1527R –PCI (Enter)

2. PC card interface (HS6000EIP01H)
   > TM1527R –PCCD (Enter)

3. LAN adapter (HS6000ELN01H)
   > TM1527R –ELN (Enter)

4. USB adapter (HS6000EIU01H or HS6000EIU02H)
   > TM1527R –USB (Enter)

The High-performance Embedded Workshop must be installed before the test program is executed.

Be sure to initiate the diagnostic program from <Drive>:\Diag\1527R. Do not initiate it from a directory other than <Drive>:\Diag\1527R, such as > <Drive>:\Diag\1527R\TM1527R –PCI (Enter). If the diagnostic program is initiated when the current directory is not <Drive>:\Diag\1527R, the diagnostic program will not operate correctly.

When –S is added to the command line, as in > TM1527R –PCI –S (Enter), steps 1 to 15 will be repeatedly executed. To stop execution in this case, enter Q.

Notes: 1. <Drive> is the drive name of the CD-ROM drive.
2. Do not remove the CD-R from the CD-ROM drive during execution of the test program.

The following messages are displayed during the test. There are 15 steps in this test.

Message	Description
`H8SX/1527R E6000H Emulator Tests V*.*` `Copyright (c) 2005 Renesas Technology Corp.`	Test program start message. x.x shows the version number.
`  Loading driver .....OK (Use PCI)`	Shows that the PC interface board is correctly installed in the host computer.
`  Initializing driver .....OK`	
`  Searching for interface card .....OK`	
`  Checking emulator is connected .....OK`	Shows that the E6000H emulator is correctly connected to the host computer.
`Emulator board information:` `    Main board ID:   H'*    Emulation board ID:  H'***`	Shows the ID number of the E6000H emulator.
`Normal started at WWW MMM DD hh:mm:ss YYYY`	Shows the time when the diagnostic program was started up.

```
******* NORMAL TEST - Press 'Q' to stop *******          (COUNT=0001)
  1. Main Board Access
     01) Registers Initial Value Check .....OK
     02) Registers Write/Verify .....OK
     03) DPRAM Address Decode Test .....SKIP
     04) DPRAM Marching Test .....SKIP
     05) Trace Memory Address Decode Test .....OK
     06) Trace Memory Marching Test .....OK
     07) G/A Registers Initial Value Check .....OK
     08) G/A Registers Write/Verify .....OK
  2. Emulation Board Access
     01) Registers Initial Value Check .....OK
     02) Registers Write/Verify .....OK
  3. Evaluation Board Access
     01) Registers Initial Value Check .....OK
     02) Registers Write/Verify .....OK
     03) H-UDI IDCODE Check .....OK
     04) Firmware BOOT .....OK
     05) Configuration Set .....OK
  4. Basic Function
     01) GO to BREAK .....OK
     02) RESET GO .....OK
     03) STEP .....OK
     04) KEYBREAK .....OK
     05) BRKCONT .....OK
     06) Internal ROM Address Decode Test .....OK
     07) Internal ROM Marching Test .....OK
     08) Internal RAM Address Decode Test .....OK
     09) Internal RAM Marching Test .....OK
     10) RESERVED
     11) RESERVED
     12) MAP Break .....OK
  5. GO to BREAK Time Measurement
     01) Counter Test Mode (EMU 12MHz MPU 12MHz Sampling 20ns) ..OK
     02) EMU 12Mhz MPU 12Mhz Sampling 20ns .....OK
     03) EMU 12Mhz MPU 12Mhz Sampling 1.6us .....OK
     04) EMU 12Mhz MPU 12Mhz Sampling 52us .....OK
     05) EMU 12Mhz MPU 12Mhz Sampling MPU .....OK
     06) EMU 12Mhz MPU 12Mhz Sampling MPU/2 .....OK
     07) EMU 12Mhz MPU 12Mhz Sampling MPU/4 .....OK
     08) EMU 12Mhz MPU 12Mhz Sampling MPU/8 .....OK
     09) EMU 12Mhz MPU 48Mhz Sampling 20ns .....OK
     10) EMU  4Mhz MPU 32Mhz Sampling 20ns .....OK
```

```
     11) EMU 35.7Mhz MPU 35.7Mhz Sampling 20ns ..................OK
     12) EMU 35.9Mhz MPU 35.9Mhz Sampling 20ns ..................OK
     13) EMU f1Mhz MPU f2Mhz Sampling 20ns ......................SKIP
  6. Emulation Monitor
     01) RESERVED
     02) RESERVED
     03) ASEST ..................................................OK
  7. G/A Break Function
     01) Address Condition ......................................OK
     02) Data Condition .........................................OK
     03) Control Signal Condition (ASEDSHH/HL/LH/HL) ............OK
     04) Function Code Condition (ASEST3-0) .....................OK
     05) RESERVED
     06) Control Signal Condition (CB22-19) .....................OK
     07) Control Signal Condition (CB18-16) .....................OK
  8. G/A Performance Analysis Function
     01) Time Measurement (20ns Sampling) .......................OK
  9. G/A Monitor Function
     01) RUN ....................................................OK
     02) VCCDOWN ................................................SKIP
     03) NOCLK ..................................................OK
     04) TIMEOUT ................................................OK
 10. G/A Parallel RAM Monitor
     01) PRAM Clear .............................................OK
     02) PRAM Monitor ...........................................OK
 11. G/A Trace Function
     01) Free Trace .............................................OK
     02) Trace Stop .............................................OK
     03) Time Stamp .............................................OK
 12. Combination
     01) B to A Time Measurement( FPGA counter ) ................OK
     02) B to A Time Measurement( G/A counter ) .................OK
     03) D to C Time Measurement( G/A counter ) .................OK
 13. Parallel Access
     01) Internal ROM Parallel Test .............................OK
     02) Internal RAM Parallel Test .............................OK
     03) Internal I/O Parallel Test .............................OK
 14. H8SX/1527 Register Read/Write
     01) Register Initial Value Check ...........................OK
     02) Register Write/Verify ..................................OK
 15. FPGA Parallel RAM Function
     01) CH0 256Byte Area Check(Byte) ......................................OK
     02) CH0 256Byte Area Check(Word) ......................................OK
     03) CH0 256Byte Area Check(Long Word) .................................OK
     04) CH1-CH11 256Byte Area Check(Long Word) ............................OK

Normal stopped at WWW MMM DD hh:mm:ss YYYY

Tests run for xh:xmin:xs

Summary:
    Tests performed 1 time(s).
  1. Main Board Access                  : 0 Error(s)
  2. Emulation Board Access             : 0 Error(s)
  3. Evaluation Board Access            : 0 Error(s)
  4. Basic Function                     : 0 Error(s)
  5. GO to BREAK Time Measurement       : 0 Error(s)
  6. Emulation Monitor                  : 0 Error(s)
  7. G/A Break Function                 : 0 Error(s)
  8. G/A Performance Analysis Function  : 0 Error(s)
  9. G/A Monitor Function               : 0 Error(s)
 10. G/A Parallel RAM Monitor           : 0 Error(s)
 11. G/A Trace Function                 : 0 Error(s)
 12. Combination                        : 0 Error(s)
 13. Parallel Access                    : 0 Error(s)
 14. H8SX/1527R Register Read/Write     : 0 Error(s)
 15. FPGA Parallel RAM Function         : 0 Error(s)
```

Shows the time when diagnostic program execution ended.

Shows the execution time of the diagnostic program.

Shows the total number of errors for each test item.

H8SX/1544 Hardware Part

RENESAS

Section 1 Overview

1.1 Notes on Usage

CAUTION

READ the following warnings before using the emulator product. Incorrect operation will damage the user system and the emulator product. The USER PROGRAM will be LOST.

1. Check all components with the component list after unpacking the emulator.
2. Never place heavy objects on the casing.
3. Observe the following conditions in the area where the emulator is to be used:
   - Make sure that the internal cooling fans on the sides of the emulator must be at least 20 cm (8") away from walls or other equipment.
   - Keep out of direct sunlight or heat. Refer to section 3.1, Environmental Conditions.
   - Use in an environment with constant temperature and humidity.
   - Protect the emulator from dust.
   - Avoid subjecting the emulator to excessive vibration. Refer to section 3.1, Environmental Conditions.
4. Protect the emulator from excessive impacts and stresses.
5. Before using the emulator's power supply, check its specifications such as power voltage and frequency.
6. When moving the emulator, take care not to subject it to strong vibration or mechanical shock.
7. After connecting the cable, check that it is connected correctly. For details, refer to section 2, Preparation before Use.
8. Supply power to the emulator and connected parts after connecting all cables. Cables must not be connected or removed while the power is on.
9. For details on differences between the target MCU and the emulator, refer to section 3.4, Support of the Target MCU.

1.2 Emulator Hardware Components

The emulator consists of an E6000H station and the E6000H's front-end unit. By installing a user system interface cable (option) to your host computer, the emulator can be connected in the same package as the device. PC interface (option) includes a PC interface board (PCI bus and PC card bus), a LAN adapter (connected with the network), and a USB adapter (connected with the USB interface). By connecting the emulator to the host computer via those interfaces, the High-performance Embedded Workshop can be used for debugging. For details on the PC interface boards (available for PCI bus and PC card bus specifications), the LAN adapter, and the USB adapter, refer to the relevant descriptive documents.

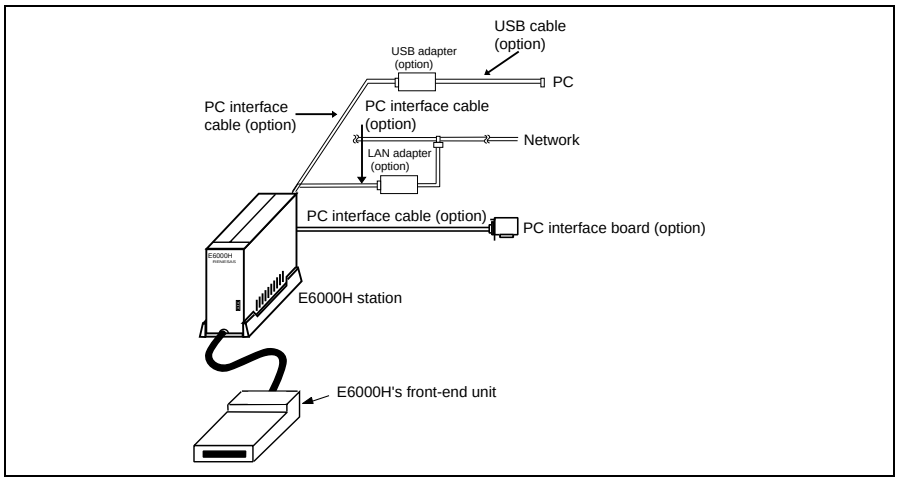


Figure 1.1 Emulator Hardware Components

1.2.1 E6000H Station Components (A Part of Photos may be Different from Real Appearances)

The names of the components on the front/rear panel of the E6000H station are listed below.

Front Panel:

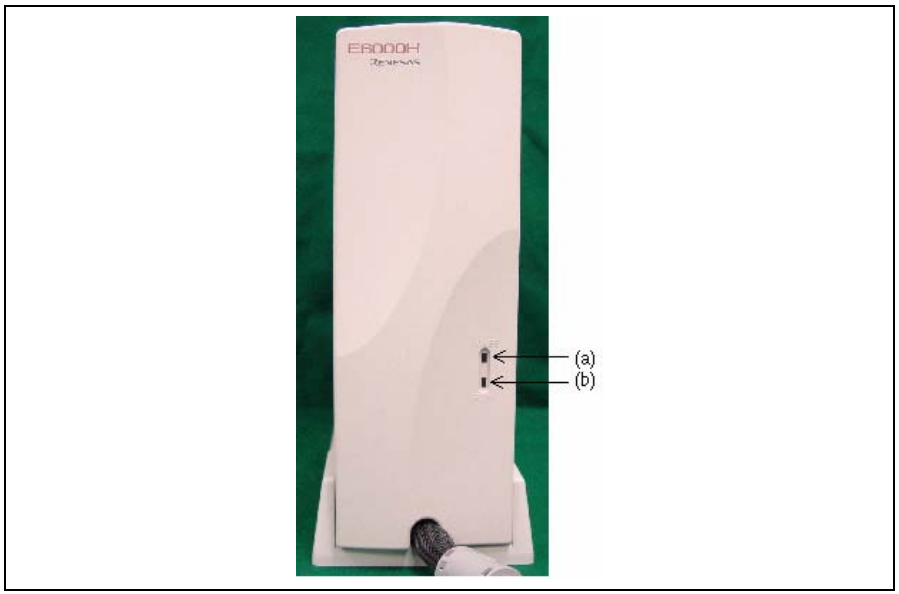


Figure 1.2 E6000H Station: Front Panel

(a) POWER lamp: Is lit up while the E6000H station is supplied with power.
(b) RUN lamp: Is lit up while the user program is running.

Rear Panel:

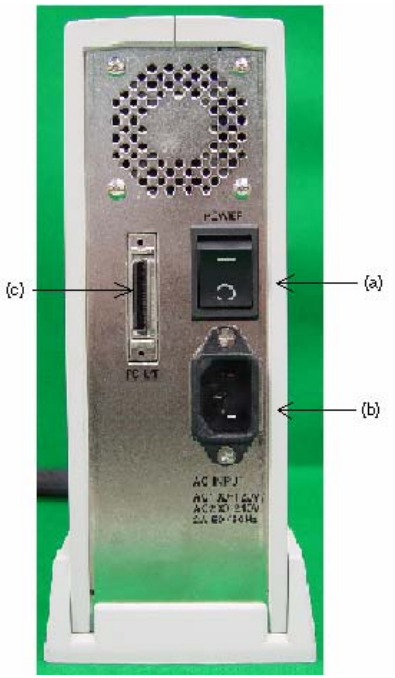


Figure 1.3 E6000H Station: Rear Panel

(a)	Power switch:	Turning this switch to I (input) supplies power to the emulator (E6000H station and the E6000H's front-end unit).
(b)	AC power connector:	For an AC 100-V to 240-V power supply.
(c)	PC interface cable connector:	For the PC interface cable that connects the host computer to the E6000H station. A PC interface board, PC card interface, LAN adapter, or USB adapter can be connected. Marked PC/IF.

1.2.2 Front-end Unit Configuration

The names of the components on the front-end unit are listed below.

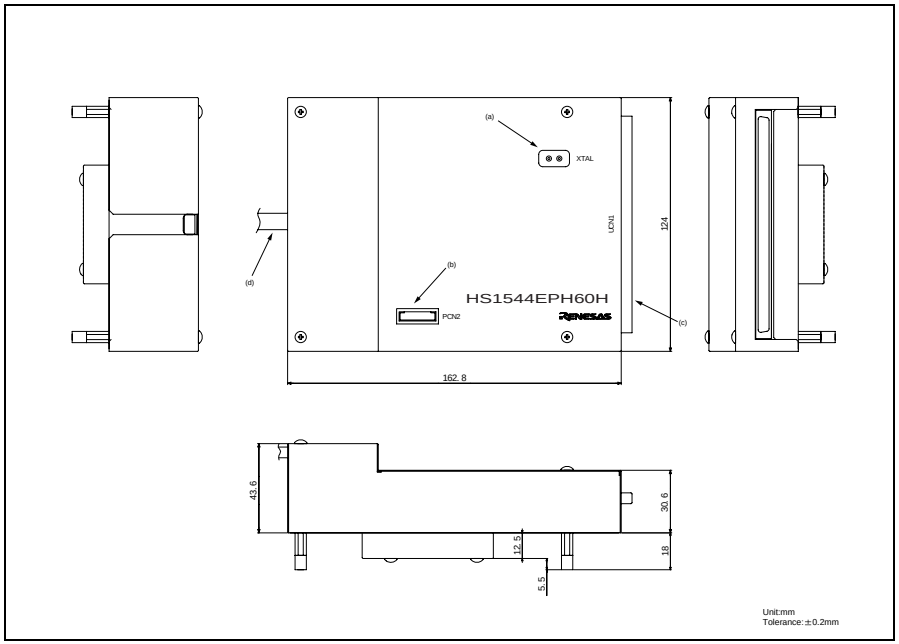


Figure 1.4 Configuration and Dimensions of the Emulator's Front-end Unit

(a) Crystal oscillator terminals:	For installing a crystal oscillator to be used as an external clock source for the MCU.
(b) External probe connector:	For connecting the external probe for external signal tracing and multibreak detection.
(c) User system interface cable connector:	For connecting the user system interface cable.
(d) Trace cable:	For connecting the front-end unit to the emulator station.

1.3 System Configuration

The emulator must be connected to a host computer via the selected PC interface board (PCI bus or PC card bus), LAN adapter, or USB adapter. For details on the PC interface boards (available for PCI bus and PC card bus specifications), the LAN adapter, and the USB adapter, refer to the relevant descriptive documents.

1.3.1 System Configuration Using Various Interfaces

(1) PC Interface Board

Figure 1.5 shows the configuration of a system in which the PC interface board is used. The emulator can be connected to a host computer via a PC interface board (option: PCI bus or PC card bus). Install the PC interface board to the expansion slot for the interface board in the host computer, and connect the interface cable supplied with the PC interface board to the emulator.

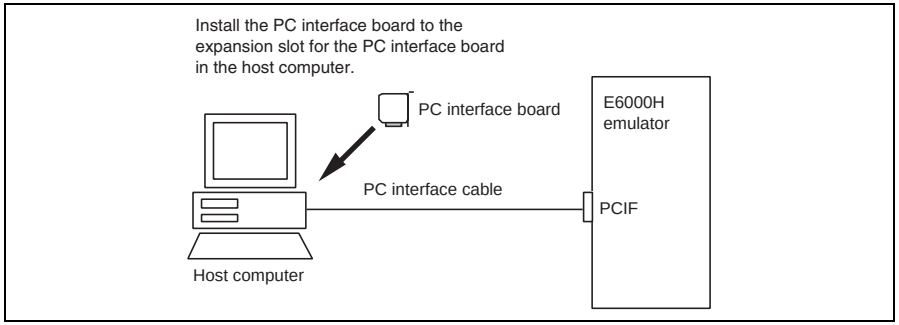


Figure 1.5 System Configuration Using a PC Interface Board

(2) LAN Adapter

Figure 1.6 shows the configuration of a system in which the LAN adapter is used. A LAN adapter can be used to connect the emulator to a host computer as a network.

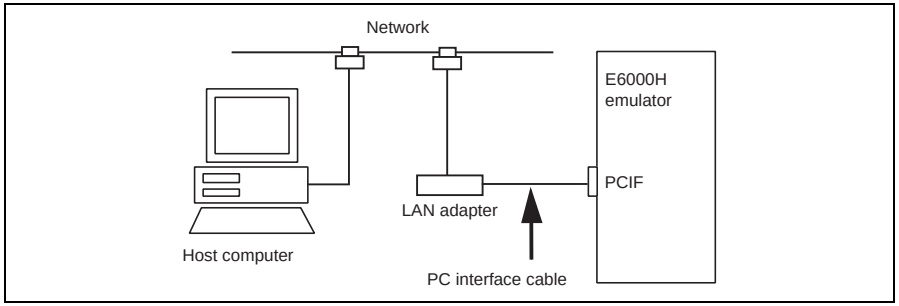


Figure 1.6 System Configuration Using a LAN Adapter

(3) USB Adapter

Figure 1.7 shows the configuration of a system in which the USB adapter is used. A USB adapter can be used to connect the emulator to a host computer with the USB interface.

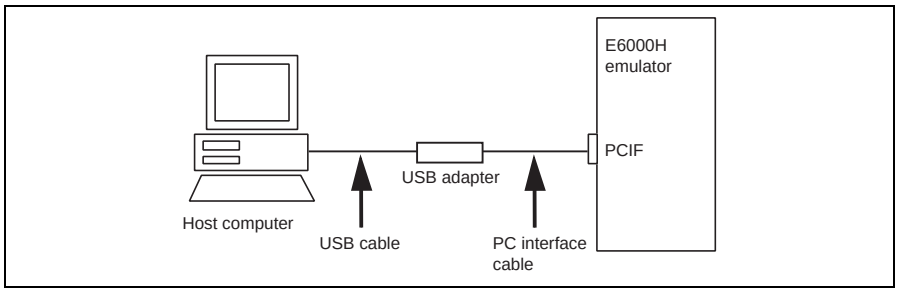


Figure 1.7 System Configuration Using a USB Adapter

Section 2 Preparation before Use

2.1 Description on Emulator Usage

This section describes the preparation before emulator usage. Figure 2.1 is a flowchart on preparation before use of the emulator.

CAUTION

Read this section and understand its contents before preparation.
Incorrect operation will damage the user system and the emulator.
The USER PROGRAM will be LOST.

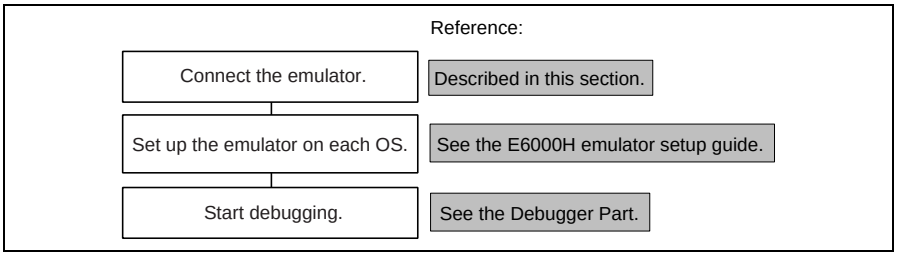


Figure 2.1 Emulator Preparation Flowchart

2.2 Emulator Connection

The following description covers connection of the emulator.

2.2.1 Connecting the Emulator to the User System

⚠WARNING

Always switch OFF the emulator and user system before connecting or disconnecting any CABLES. Failure to do so will result in a FIRE HAZARD and will damage the user system and the emulator or will result in PERSONAL INJURY. The USER PROGRAM will be LOST.

1. Check that the emulator power switch is turned off. Ensure that the power lamp on the right side of the E6000H station's front panel is not lit.
2. Remove the AC power cable of the E6000H station from the outlet (if the cable is connected to the outlet).
3. The emulator is connected to the user system via the user system interface cable.

2.2.2 Connecting the User System Interface Cable

⚠WARNING

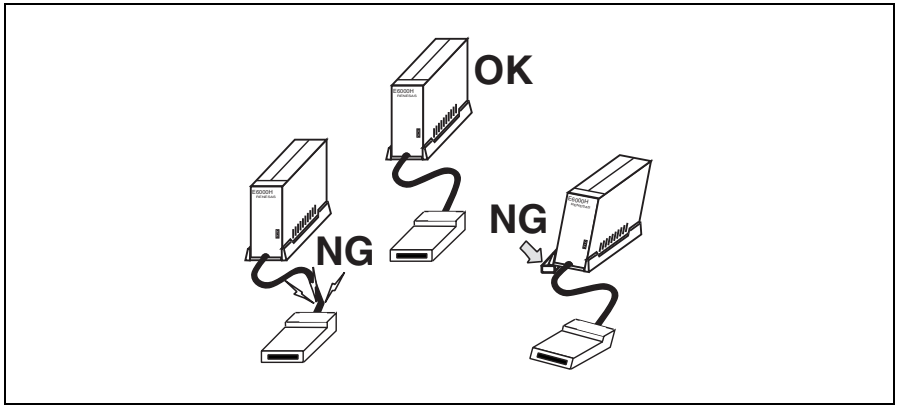
Always switch OFF the emulator and user system and check pin numbers on the connectors and IC socket before connecting or disconnecting the USER SYSTEM INTERFACE CABLE. Connection with the power on or incorrect connection will damage the emulator, user system interface cable, and user system, and result in a FIRE HAZARD.

For details on the method of connecting the user system interface cable, refer to the descriptions of the user system interface cables for individual H8SX/1544 E6000H-series products.

2.2.3 Precautions on Connecting the User System

When connecting the emulator to the user system, note the following:

1. Secure the E6000H station location.
   Place the E6000H station and the E6000H's front-end unit so that the trace cable is not bent or twisted, as shown below. A bent or twisted cable will impose stress on the user interface, leading to connection or contact failure. Make sure that the E6000H station and the front-end unit are placed in a secure position so that they do not move and impose stress on the user interface while being used.



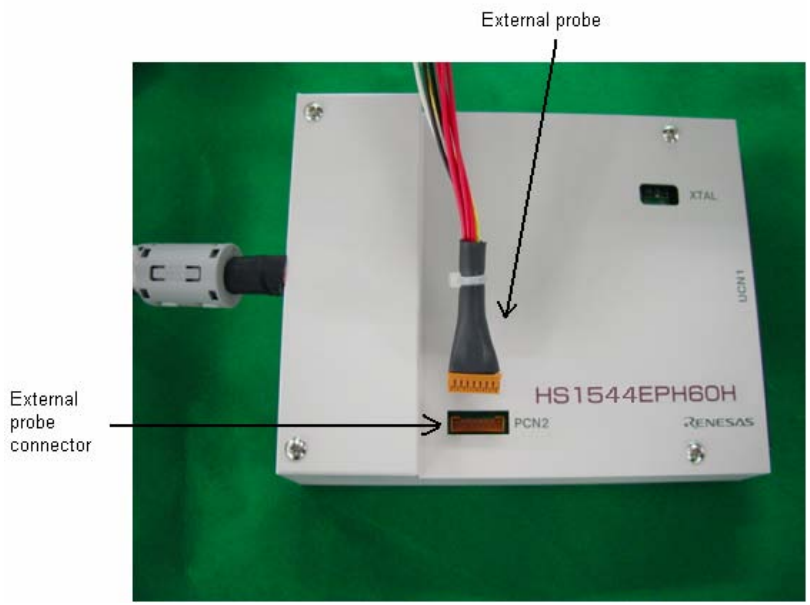
2. Make sure the power supply is off.
   Before connecting the emulator to the user system, check that the emulator and the user system are turned off.
3. Connect Vcc to the user system power.
   The emulator monitors and determines whether the user system is turned on or off by the Vcc pins.
   After connecting the user system to the emulator via the user system interface cable, be sure to supply power to the Vcc pins. Otherwise, the emulator assumes that the user system is not connected.
   When the user system is connected, check that the power of the user system is supplied to these pins.

2.2.4 Connecting the External Probe

CAUTION

Check the external probe direction and connect the external probe to the emulator station correctly. Incorrect connection will damage the probe or connector.

When an external probe is connected to the external probe connector on the emulator's front-end unit, it enables external signal tracing and multibreak detection. Figure 2.2 shows the external probe connector.



Pin No.	Probe Name	Signal Name	Note
1	1	Probe input 0	Synchronous break input pin
2	2	Probe input 1	
3	3	Probe input 2	
4	4	Probe input 3	
5	G	GND	GND connection pin
6	TUV	Trigger output	Trigger mode output pin
7	G	GND	GND connection pin
8	T5V	RUN or break output	High-level output at RUNSTEP

Figure 2.2 External Probe Connector

2.2.5 Selecting the Clock

This emulator supports three types of clock for the MCU: a crystal oscillator attached on the emulator's front-end unit, external clock input from the user system, and the emulator internal clock. The clock is specified with the [Configuration] dialog box.

This emulator can use a clock source (ϕ) running at up to 40.0 MHz as the H8SX/1544 clock input.

CLOCK command ——— Xtal (Crystal oscillator: 4 to 9 MHz)

Target (External clock: 4 to 9 MHz)

4 (Emulator internal clock: 4 MHz)

5 (Emulator internal clock: 5 MHz)

Crystal Oscillator: A crystal oscillator is not supplied with the emulator. Prepare and use one that has the same frequency as that of the user system. When using a crystal oscillator as the MCU clock source, the frequency range must be from 4 to 9 MHz.

CAUTION

Always switch OFF the emulator and user system before connecting or disconnecting the CRYSTAL OSCILLATOR. Otherwise, the USER PROGRAM will be LOST.

Follow the procedure listed below to install the crystal oscillator:

1. Check that the emulator power switch is turned off. (Check that the power LED is not lit.)
2. Attach the crystal oscillator into the terminals on the emulator's front-end unit (figure 2.3). Be careful to ensure that the terminals do not touch the casing.
3. Turn on the user system power and then the emulator power. The crystal oscillator will then be automatically set and started up. This function will allow the execution of the user program at the operating frequency of the user system even when the user system is not connected to the emulator.

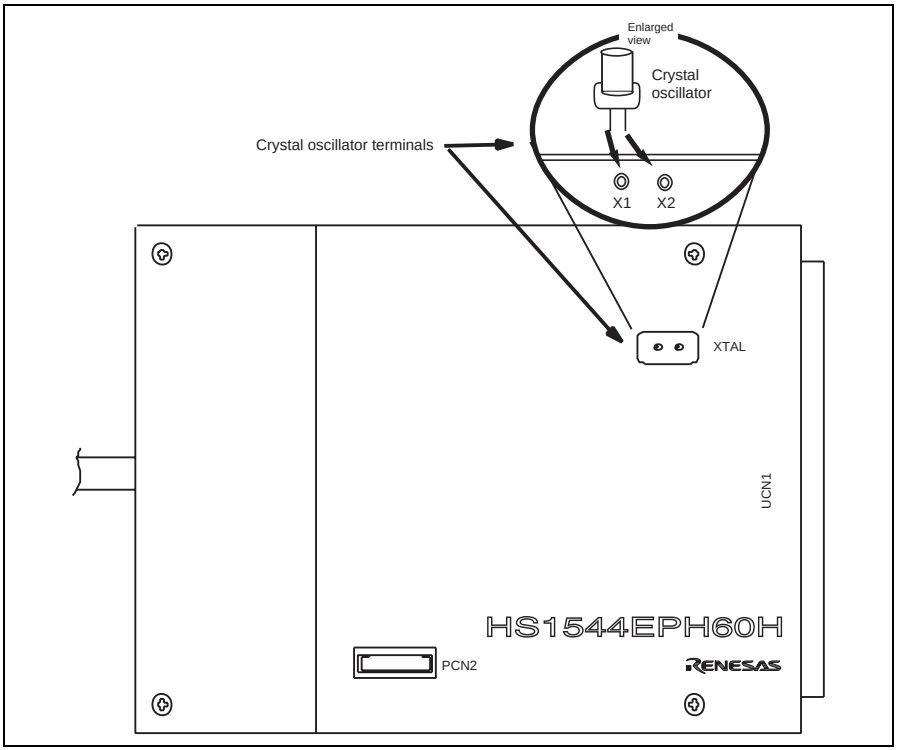


Figure 2.3 Installing the Crystal Oscillator

External Clock: Follow the procedure listed below to select the external clock.

1. Check that the emulator power switch is turned off. (Check that the power LED is not lit.)
2. Connect the user system interface cable to the user system and supply a clock signal through the EXTAL pin from the user system.
3. Turn on the user system power and then the emulator power. The external clock source will then be automatically specified.

Emulator Internal Clock: Specify 4.0 MHz or 5.0 MHz in the [Configuration] dialog box.

Reference:

When the emulator system program is initiated, the emulator automatically selects the MCU clock source according to the following priority:

1. User system's clock when an external clock is supplied from the user system
2. Crystal oscillator, if one is mounted on the emulator's front-end unit
3. Emulator-internal clock

2.2.6 Connecting the System Ground

CAUTION

Separate the frame ground from the signal ground at the user system. When the frame ground is connected to the signal ground and the emulator is then connected to the user system, the emulator will malfunction.

The emulator's signal ground is connected to the user system's signal ground via the emulator's front-end unit. In the E6000H station, the signal ground and frame ground are connected (figure 2.4). At the user system, connect the frame ground only; do not connect the signal ground to the frame ground.

If it is difficult to separate the frame ground from the signal ground in the user system, ground the frame to the same outlet as the 100-V to 240-V AC power supply of the emulator station (figure 2.5) so that the ground potentials become even.

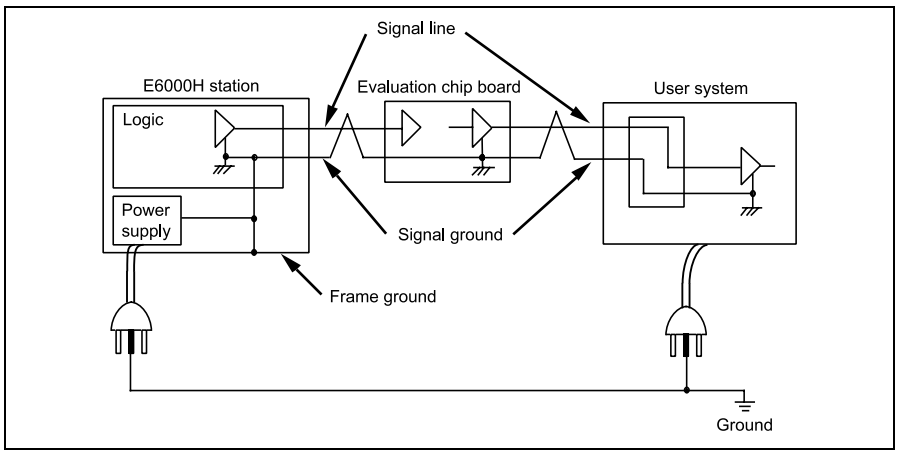


Figure 2.4 Connecting the System Ground

WARNING

Always switch OFF the emulator and user system before connecting or disconnecting any CABLES. Failure to do so will result in a FIRE HAZARD and will damage the user system and the emulator or will result in PERSONAL INJURY. The USER PROGRAM will be LOST.

The user system must be connected to an appropriate ground so as to minimize noise and the adverse effects of ground loops. When connecting the emulator's front-end unit and the user system, confirm that the ground pins of the front-end unit are firmly connected to the user system's ground.

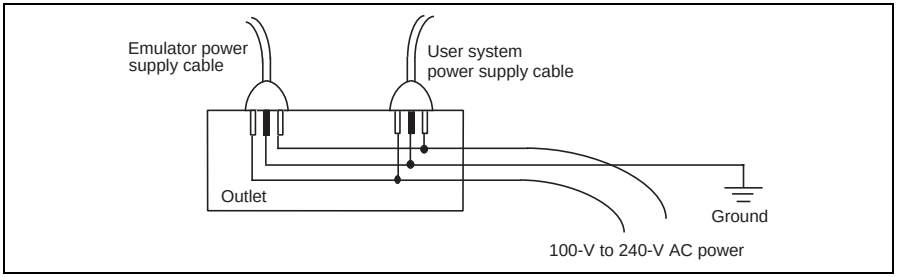


Figure 2.5 Connecting the Frame Ground

2.2.7 PC Interface Board Specifications

For details on the PC interface board, LAN adapter, or USB adapter, refer to their description notes.

Section 3 Hardware Specifications

3.1 Environmental Conditions

CAUTION

Observe the conditions listed in table 3.1 when using the emulator. The following environmental conditions must be satisfied, otherwise the user system and the emulator will not operate normally. The USER PROGRAM will be LOST.

Table 3.1 Environmental Conditions

Item	Specifications	
Temperature	Operating:	+10 to +35°C
	Storage:	−10 to +50°C
Humidity	Operating:	35 to 80% RH, no condensation
	Storage:	35 to 80% RH, no condensation
Vibration	Operating:	2.45 m/s^2 max.
	Storage:	4.9 m/s^2 max.
	Transportation:	14.7 m/s^2 max.
AC input power	Voltage:	100 V to 240 V AC
	Frequency:	50/60 Hz
	Power consumption:	75 W
Ambient gases	There must be no corrosive gases present.	

3.2 Emulator External Dimensions and Mass

Figures 3.1 shows the external dimensions and mass of the E6000H emulator.

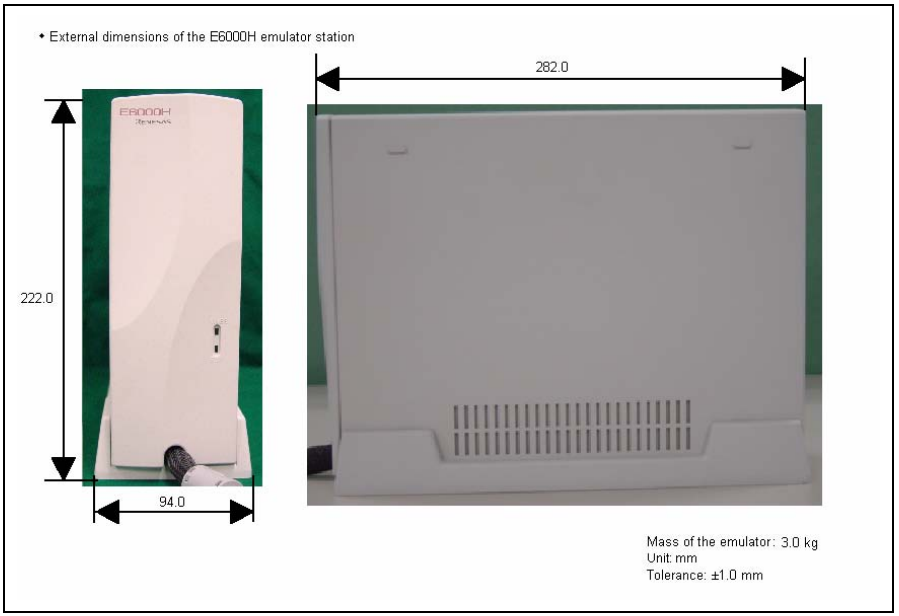


Figure 3.1 External Dimensions and Mass of the Emulator

3.3 User System Interface Circuit

3.3.1 User System Interface Circuit

The circuits that interface the MCU in the emulator to the user system include the level-shifter circuits (for 5 V and 3.3 V) and resistors. When connecting the emulator to a user system, adjust the user system hardware compensating for FANIN, FANOUT, and propagation delays.

The user system interface circuits connected to the user system are shown in figure 3.2.

The delay time is generated on the timing of the _RES, _NMI, and _STBY signals when they are input to the MCU from the user system, as shown in table 3.2, because this connection for those signals is via logic circuit on the emulator's front-end unit.

Table 3.2 Delay Time for Signal Connected via the Emulator's Front-end Unit

Signal Name	Delay Time (ns)
_RES	27
_NMI	27
_STBY	27

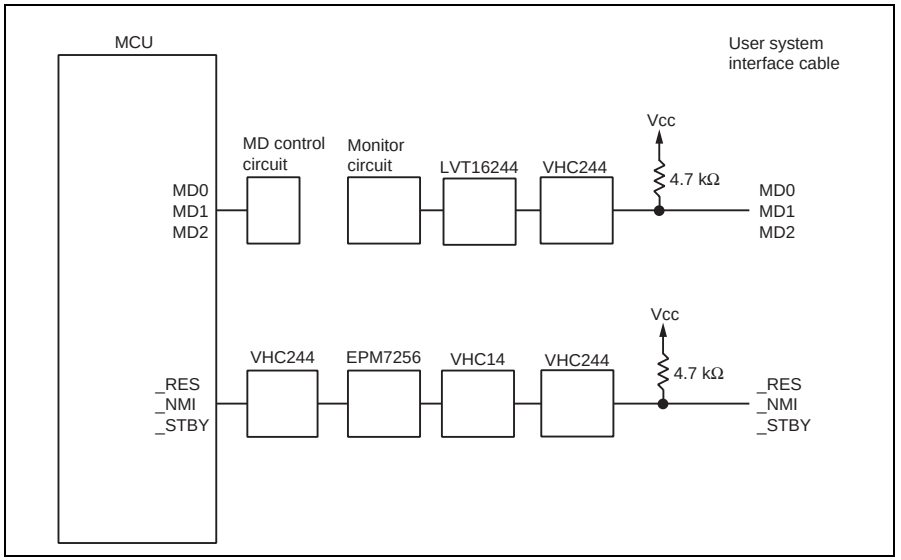


Figure 3.2 User System Interface Circuits (1)

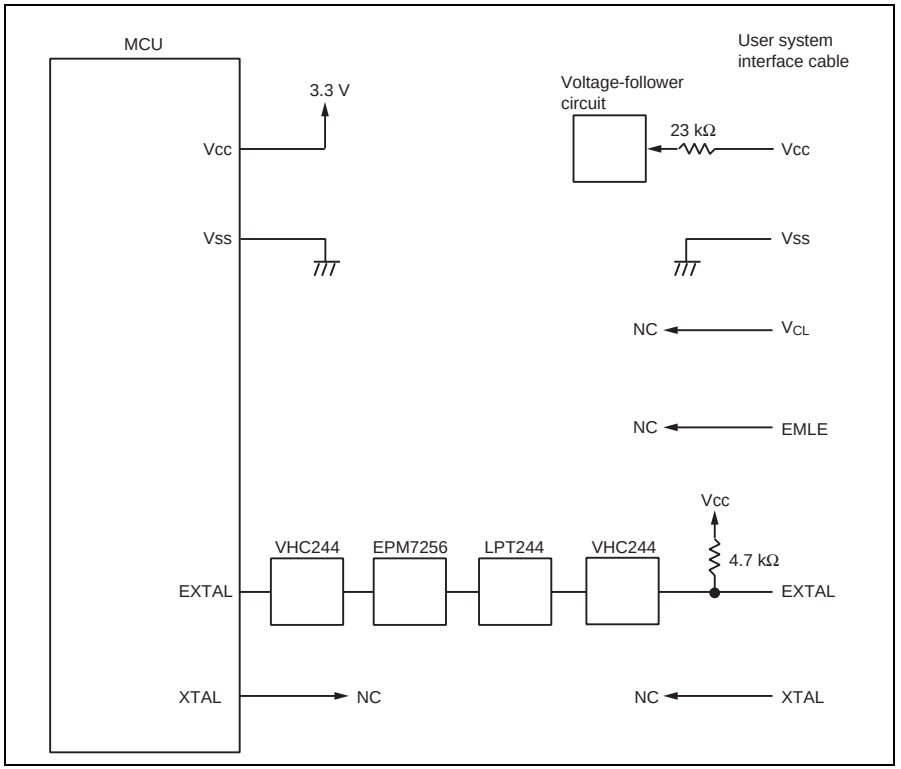


Figure 3.2 User System Interface Circuits (2)

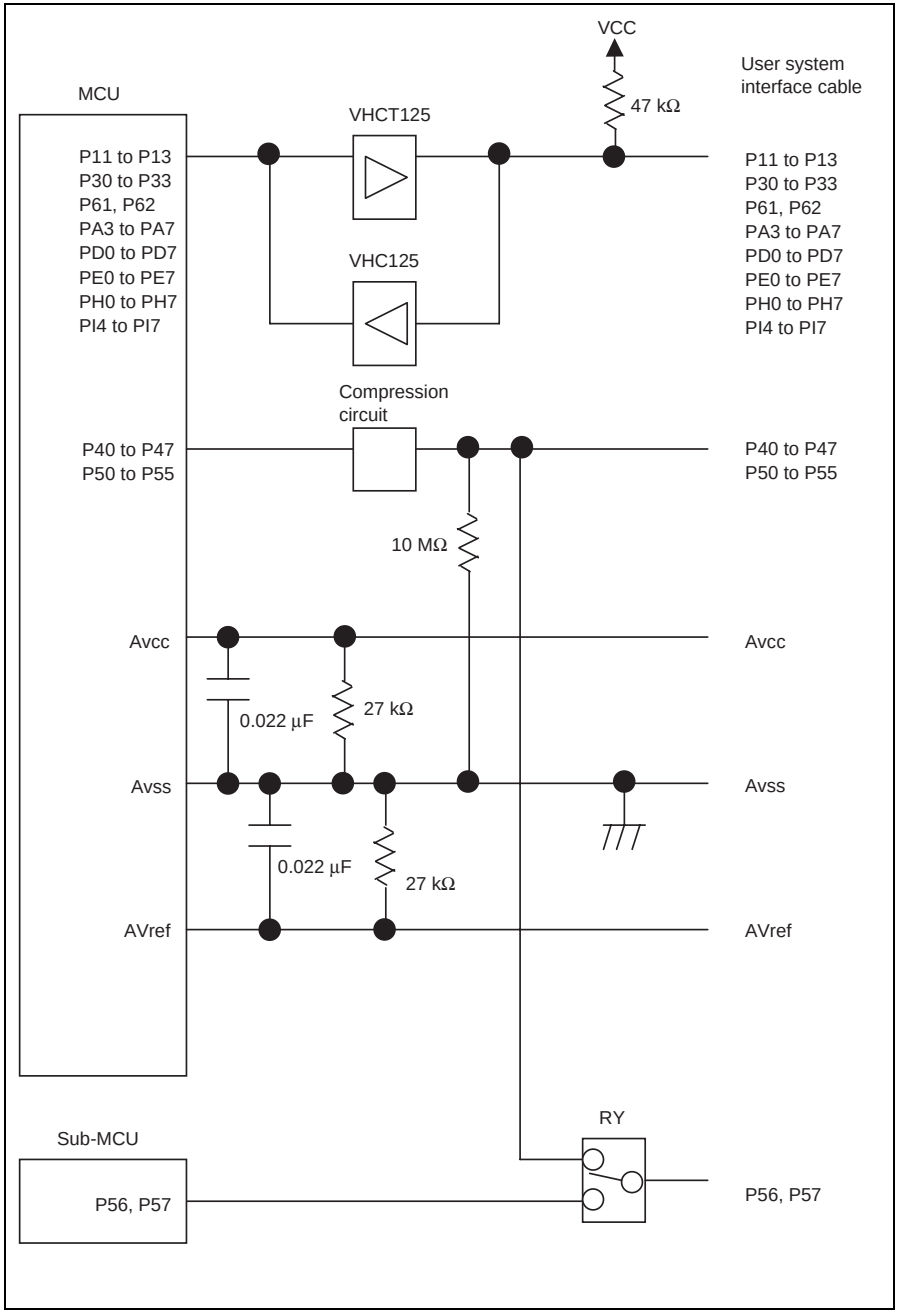


Figure 3.2 User System Interface Circuits (3)

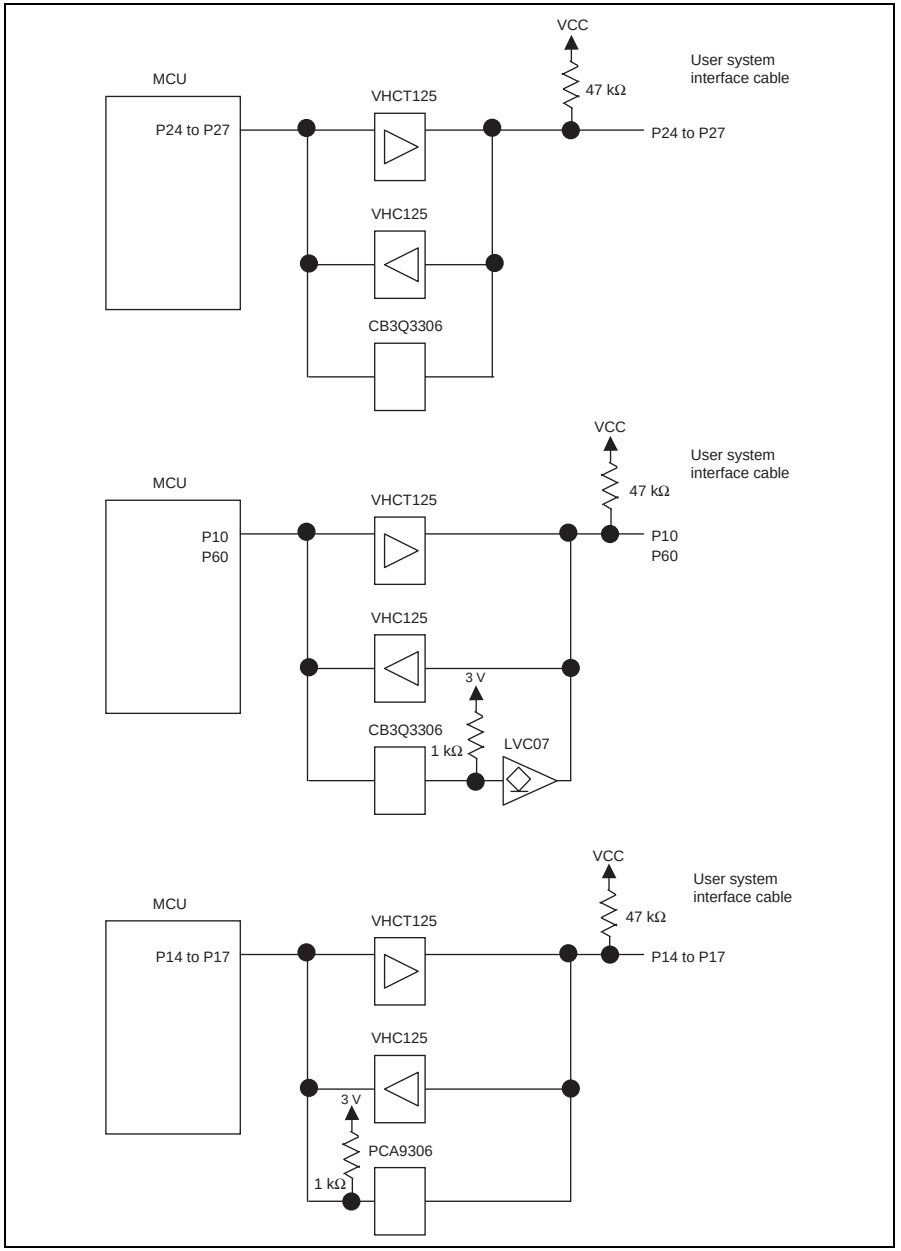


Figure 3.2 User System Interface Circuits (4)

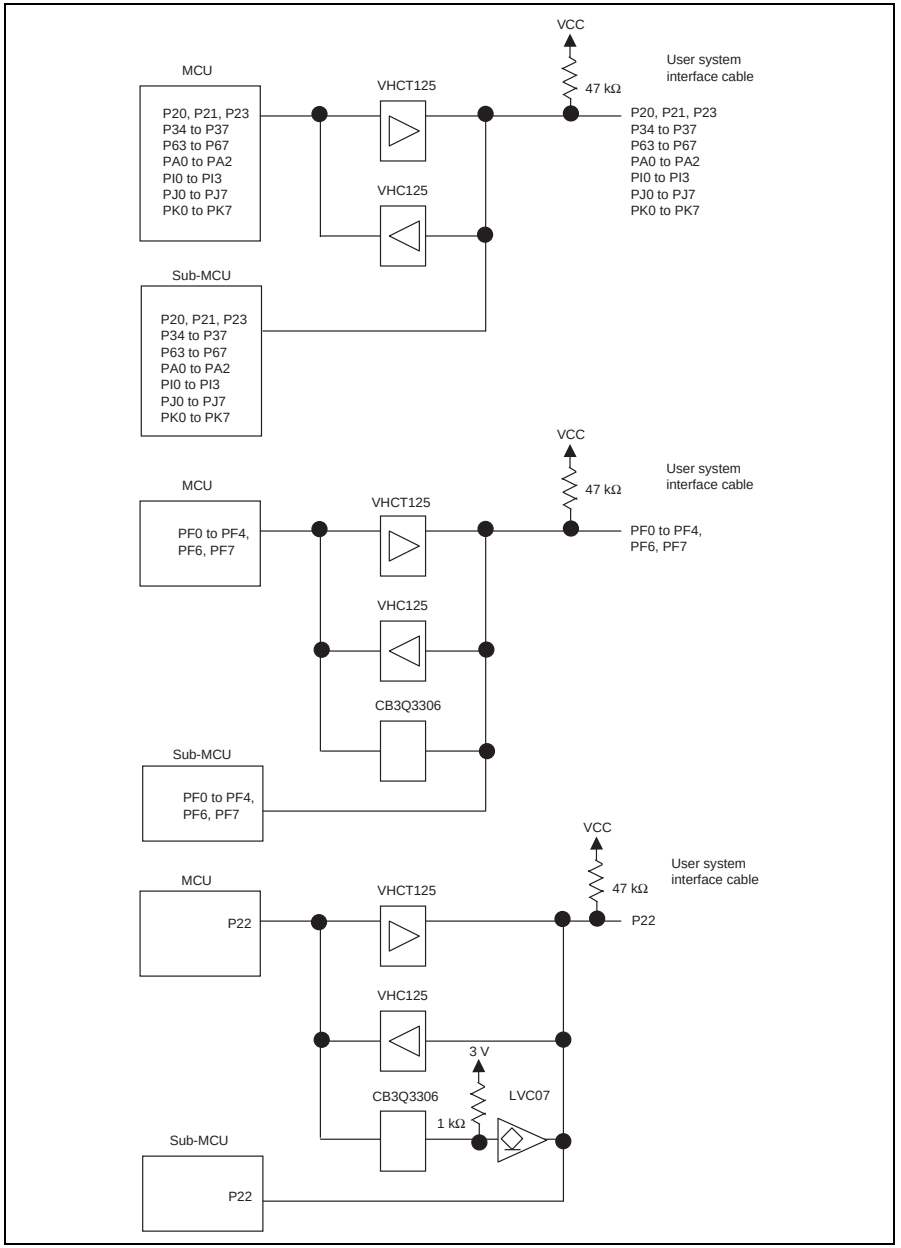


Figure 3.2 User System Interface Circuits (5)

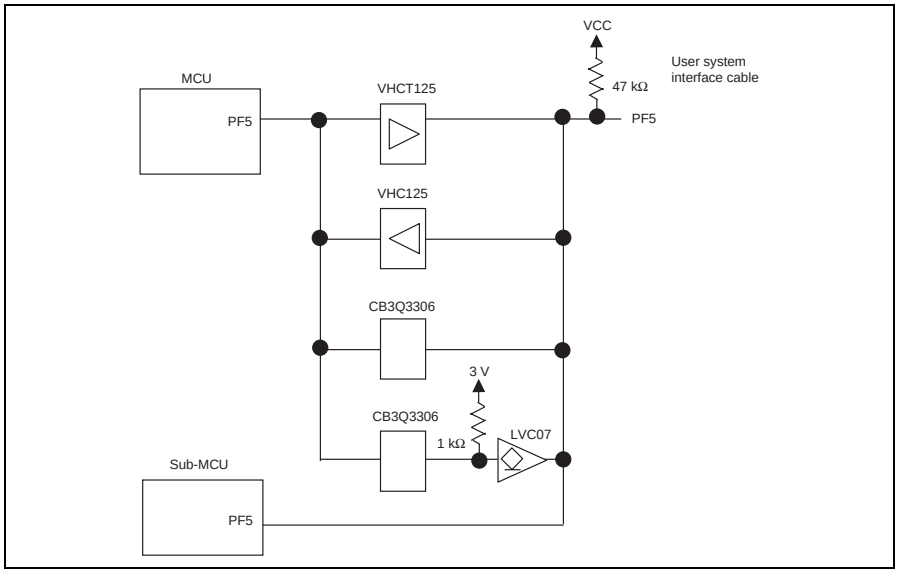


Figure 3.2 User System Interface Circuits (6)

3.4 Support of the Target MCU

3.4.1 Memory Space

The architecture of the MCU allows for a 16-Mbyte memory space.

(1) On-Chip I/O Area

If an attempt is made to access the on-chip I/O area, the on-chip I/O area in the MCU installed in the emulator is accessed. To break user program execution when the on-chip I/O area is written to or accessed, use the hardware break or internal break.

(2) On-Chip ROM/RAM Area

If an attempt is made to access the on-chip ROM/RAM area, the on-chip ROM/RAM area in the MCU installed in the emulator is accessed. To break user program execution when the on-chip ROM/RAM area is written to or accessed, use the hardware break or internal break.

3.4.2 Power-Down Modes (Sleep, Software Standby, Hardware Standby, and All-Module-Clock-Stop)

For reduced power consumption, the MCU has sleep, software standby, hardware standby, and all-module-clock-stop modes.

(1) Sleep and Software Standby Modes

- Break

  The MCU can be taken out of the sleep and software standby modes either in the normal ways or through satisfaction of a break condition (forced break). When restarting after a break, the user program will restart at the instruction following the SLEEP instruction.
- Trace

  Trace information is not acquired in these modes.
- Memory access with emulator functions

  For information on displaying and modifying the contents of memory in the sleep and software standby modes, refer to section 5.4, Displaying and Modifying the Contents of Memory, in the Debugger Part.

3.4.3 Interrupts

During execution and step execution, the user can interrupt the MCU.

Interrupt sources are retained while emulation is halted (break mode). In such cases, interrupt processing commences immediately after emulation is restarted.

3.4.4 Control Input Signals (/RES, /NMI, and /STBY)

The MCU control input signals are /RES, /NMI, and /STBY. The /RES, /NMI, and /STBY signals are only valid when emulation was started with normal program execution (i.e., they are invalid when emulation was started with step execution).

The input of the /RES, /NMI, or /STBY signal during execution or step execution can be disabled by a setting in the [Configuration] dialog box.

3.4.5 Watchdog Timer (WDT)

When emulation is suspended (i.e. by a break), counting up by the WDT timer counter (TCNT) is also suspended. Counting resumes when emulation is resumed (user mode).

During break mode, a prescaler, which supplies a clock to TCNT, operates continuously. Since the prescaler might be in different phases before and after emulation goes through a period in the break mode, a break can change the period before the WDT overflows by ±1 cycle of the prescaler's clock.

3.4.6 A/D Converter

As well as analog input pins, the A/D converter has AVcc, AVss, Avref, and /ADTRG pins. As the A/D converter operates with an independent power supply, connect AVcc (the power supply pin) to the A/D power supply on the user system.

Notes:
1. When the A/D converter is not in use, connect AVcc to Vcc.
2. As there are user system interface cable, compression circuits (for compression at the ratio of 5:3.3), wiring on the printed circuit board, and protective circuits are connected between the user system and the MCU on the evaluation chip board, the precision of conversion is lower than that of the actual MCU.

3.4.7 Emulator State and Internal Modules

Operation of some internal modules depends on the emulator's state. Table 3.3 shows the relation between the emulator's state and operation of the internal modules.

Table 3.3 Emulator's State and Operation of Internal Modules

Internal Module	Operation while Emulation is Halted (Break)	Operation During Emulation (Execution or Step Execution)
DMAC (DMA controller)	Yes	Yes
TPU (16-bit timer pulse unit)	Yes	Yes
WDT (watchdog timer)	No	Yes
SCI (serial communication interface)	Yes	Yes
A/D converter	Yes	Yes
RCAN (controller area network)	Yes	Yes
SSU (synchronous serial communication unit)	Yes	Yes
Motor-control PWM	Yes	Yes
16-bit PWM	Yes	Yes
SDG (sound generator)	Yes	Yes
D/A converter	Yes	Yes
WAT (watch timer)	Yes	Yes
I/O port	Yes	Yes
H-UDI (user debugging interface)	Not available*	Not available*

Note: The user cannot use the H-UDI because it is being used by the emulator.

3.4.8 Differences in Values of Registers

Note that certain general and control registers of both emulators are initialized whenever the system is activated or the MCU is reset by a command.

Table 3.4 Initial Values of Registers in the MCU and the Emulator

	Emulator		
Register Name	**Power On**	**Reset (Reset CPU)**	**MCU (Power-On Reset)**
PC	PC value indicated by the power-on reset vector	PC value indicated by the power-on reset vector	Undefined
ER0 to ER7	H'00000000	Value before the reset	Undefined
CCR	B'1XXXXXXX	B'1XXXXXXX	B'1XXXXXXX
EXR	B'01111111	Value before the reset	B'01111111
MACH	H'00000000	Value before the reset	Undefined
MACL	H'00000000	Value before the reset	Undefined
VBR	H'00000000	Value before the reset	H'00000000
SBR	H'FFFFFF00	Value before the reset	H'FFFFFF00

Note: X indicates an undefined bit.

3.5 Notes Specific to the H8SX/1544 E6000H Emulator

3.5.1 Custom Device Function

The maximum value selectable for the on-chip ROM with the custom device functions of the H8SX/1544 E6000H emulator is 2 Mbytes. Do not specify a value greater than 2 Mbytes.

3.5.2 Subclock Operation

This emulator does not support subclock operation. Thus the following function and register cannot be used:

- WAT
- Subclock control register (SUBCKCR)

3.5.3 Open-Drain Control Registers (PnODR) —Restriction

In the emulator, the NMOS open-drain output is disabled regardless of the setting of the bits, while setting a bit of PnODR corresponding to TIOCB4, TIOCA4, TIOCA5, TIOCB5, A23 to A16, TxD5, or SCK5 as 1 enables an NMOS open-drain output in the actual MCU. Note, however, that P24 to P27 and PF0 to PF7 can be used as NMOS open-drain outputs.

3.5.4 Synchronous Serial Communication Unit (SSU) —Restriction

If a conflict error occurs while the values of bits MSS and SCKS in an SSCRH register are 1, the MSS bit will be cleared and the corresponding SSCK pin will be switched to act as an input. After that, even if the user sets the SCKS bit to 0 to reconfigure the SSCK pin to act as an I/O port pin, the emulator will not set the SSCK pin up as an I/O port pin. To allow use of the SSCK pin as an I/O port pin, have the program write 0 to the MSS bit if a conflict error occurs. This procedure is not necessary unless the SSCK pin is to be used as an I/O port pin.

3.5.5 Port Registers (PORTx) —Restriction

(1) When the module pins are used as inputs

In the emulator, undefined values are read from PORTx corresponding to the modules. In the actual MCU, however, reading PORTx allows reading the states of the pins.

(2) When the module pins are used as outputs

In the emulator, undefined values are read from PORTx corresponding to the modules when the value of the data direction registers is 0. In the actual MCU, however, reading PORTx allows reading the states of the pins.

Peripheral modules that apply: Motor-control PWM timer, 16-bit PWM, RCAN, SSU, SDG, and DA

3.5.6 Input Buffer Control Register (PnICR) —Restriction

In the emulator, pins can be used as inputs for the peripheral modules mentioned above regardless of the setting of the corresponding ICR bits.

In the actual MCU, however, the input buffers are invalid and the input signals are fixed high for pins where the corresponding ICR bit has been cleared to 0. A pin cannot be used as an input for a peripheral module unless the corresponding ICR bit is set to 1.

If you are using any of these modules, ensure that your program sets the ICR bits for the corresponding pins to 1 before the module is used.

Peripheral modules that apply: SSU and RCAN

3.5.7 Watch Timer (WAT) —Restriction

In the emulator, resetting the WDT initializes the WAT registers. In the actual MCU, the WAT registers are not initialized.

3.5.8 Port Function Control Register 4 (PFCR4) —Note

In the emulator, the value of PFCR4 is H'1F in the on-chip ROM disabled mode and A23 to A21 are not output as addresses. In the MCU, however, the value of PFCR4 is H'FF in the on-chip ROM disabled mode. To use A23 to A21 in the emulator, the user program must write the value H'FF to address H'FFFFFBC4.

3.5.9 Access to the Internal RAM —Note

In the emulator, writing to the internal RAM takes one cycle by default. To make the emulator write data to the RAM in two cycles, the user program must write the value H'35 to address H'FFFFFD97 before any RAM access. After that, the RAM should not be accessed for at least one cycle. This address will be reinitialized whenever the reset pin is activated (goes low). In such cases, write the value H'35 to the same address again after the system has been released from the reset state.

3.5.10 External Expanded Mode —Restrictions

1. When the frequency of Bϕ (external bus clock) is 20 MHz or higher, external accesses must be made in 3 or more state cycles.
2. While the emulator is operating in the external expanded mode, disabling output of Bϕ prevents external accesses.
3. The emulator starts output of the write data at the rising edge of the T2 cycle. The actual MCU, however, starts output of data at the falling edge of the T1 cycle.

Section 4 Diagnostic Test Procedure

This section describes the diagnostic test procedure using the E6000H diagnostic program.

4.1 System Set-Up for Diagnostic Program Execution

To execute the diagnostic program, use the following hardware; do not connect the user system interface board and user system.

- E6000H (HS1544EPH60H)
- Host computer
- The E6000 PC interface board which will be one of the following boards:
  PCI bus interface board (HS6000EIC01H or HS6000EIC02H)
  PC card interface (HS6000EIP01H)
  LAN adapter (HS6000ELN01H)
  USB adapter (HS6000EIU01H or HS6000EIU02H)

1. Install the E6000 PC interface board in the host computer and connect the supplied PC interface cable to the board.
2. Connect the PC interface cable to the emulator.
3. Connect the supplied AC power cable to the emulator.
4. Boot the host computer up and enter the command input wait state of DOS (the DOS prompt) (Windows® 98SE or Windows® Me) or the command prompt (Windows NT®, Windows® 2000, or Windows® XP). If the prompt window is not set up to be displayed over the whole screen, press the [Alt + Enter] key chord to switch the mode. To return the prompt to a window, press the [Alt + Enter] key. This key chord works in the same way regardless of the OS being used.

Note: If the display of the MS-DOS prompt does not switch from full-screen to windowed or vice-versa in response to the [Alt + Enter] key chord, mark the [Alt+Enter] check box in [Windows shortcut keys] of the [Misc] page on the [MS-DOS Prompt Properties] dialog box and click the [Apply] button to update the settings.

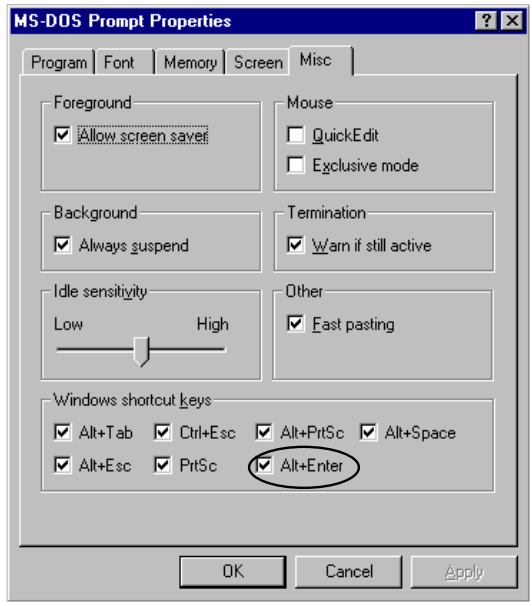


Figure 4.1 [Misc] Page

5. Turn on the E6000H emulator switch.

Note: To execute the diagnostic program, firstly turn on the power of the emulator. The diagnostic program checks the initial state of the hardware. Therefore, after turning on the power, do not activate the High-performance Embedded Workshop before executing the diagnostic program.

4.2 Test Item of the Diagnostic Program

Table 4.1 shows the test items of this diagnostic program.

Table 4.1 Test Items of the Diagnostic Program

Test No.	Test Item	Description
1	Main Board Access	Register test in the E6000H main board
2	Emulation Board Access	Register test in the E6000H emulation board
3	Evaluation Board Access	Register test in the E6000H's front-end unit
4	Basic Function	Test of the basic functions
5	GO to BREAK Time Measurement	Test of the execution time measurement function
6	Emulation Monitor	Test of the emulation monitor
7	G/A Break Function	Test of the G/A break function
8	G/A Performance Analysis Function	Test of the G/A performance measurement function
9	G/A Monitor Function	Test of the G/A monitoring function
10	G/A Parallel RAM Monitor	Test of the G/A parallel RAM monitoring function
11	G/A Trace Function	Test of the G/A trace function
12	Combination	Test of several functions in combination
13	Parallel Access	Test of the parallel access function
14	H8SX/1544 Register Read/Write	Test of registers in the SUB MCU
15	FPGA Parallel RAM Function	Test of the FPGA's parallel RAM monitoring function

4.3 Diagnostic Test Procedure Using the Diagnostic Program

Insert the CD-R (HS1650EPH60SR supplied with the emulator) into the CD-ROM drive of the host computer, use the command prompt to change the current directory to <Drive>:\Diag\1544, and enter the command below that corresponds to the PC interface board used to initiate the diagnostic program:

1. PCI bus interface board (HS6000EIC01H or HS6000EIC02H)
   > TM1544 –PCI (Enter)

2. PC card interface (HS6000EIP01H)
   > TM1544 –PCCD (Enter)

3. LAN adapter (HS6000ELN01H)
   > TM1544 –ELN (Enter)

4. USB adapter (HS6000EIU01H or HS6000EIU02H)
   > TM1544 –USB (Enter)

The High-performance Embedded Workshop must be installed before the test program is executed.

Be sure to initiate the diagnostic program from <Drive>:\Diag\1544. Do not initiate it from a directory other than <Drive>:\Diag\1544, such as > <Drive>:\Diag\1544\TM1544 –PCI (Enter). If the diagnostic program is initiated when the current directory is not <Drive>:\Diag\1544, the diagnostic program will not operate correctly.

When –S is added to the command line, as in > TM1544 –PCI –S (Enter), steps 1 to 15 will be repeatedly executed. To stop execution in this case, enter Q.

Notes: 1. <Drive> is the drive name of the CD-ROM drive.
2. Do not remove the CD-R from the CD-ROM drive during execution of the test program.

The following messages are displayed during the test. There are 15 steps in this test.

Message	Description
H8SX/1544 E6000H Emulator Tests V*.* Copyright (c) 2003 Renesas Technology Corp.	Test program start message. x.x shows the version number.
Loading driver .....OK (Use PCI)	Shows that the PC interface board is correctly installed in the host computer.
Initializing driver .....OK	
Searching for interface card .....OK	
Checking emulator is connected .....OK	Shows that the E6000H emulator is correctly connected to the host computer.
Emulator board information: Main board ID: H'* Emulation board ID: H'***	Shows the ID number of the E6000H emulator.
Normal started at WWW MMM DD hh:mm:ss YYYY	Shows the time when the diagnostic program was started up.

```
******* NORMAL TEST - Press 'Q' to stop *******          (COUNT=0001)

  1. Main Board Access
     01) Registers Initial Value Check .....OK
     02) Registers Write/Verify .....OK
     03) DPRAM Address Decode Test .....SKIP
     04) DPRAM Marching Test .....SKIP
     05) Trace Memory Address Decode Test .....OK
     06) Trace Memory Marching Test .....OK
     07) G/A Registers Initial Value Check .....OK
     08) G/A Registers Write/Verify .....OK
  2. Emulation Board Access
     01) Registers Initial Value Check .....OK
     02) Registers Write/Verify .....OK
  3. Evaluation Board Access
     01) Registers Initial Value Check .....OK
     02) Registers Write/Verify .....OK
     03) H-UDI IDCODE Check .....OK
     04) Firmware BOOT .....OK
     05) Configuration Set .....OK
  4. Basic Function
     01) GO to BREAK .....OK
     02) RESET GO .....OK
     03) STEP .....OK
     04) KEYBREAK .....OK
     05) BRKCONT .....OK
     06) Internal ROM Address Decode Test .....OK
     07) Internal ROM Marching .....OK
     08) Internal RAM Address Decode Test .....OK
     09) Internal RAM Marching .....OK
     10) Emulation RAM Address Decode Test .....SKIP
     11) Emulation RAM Marching Test .....SKIP
     12) MAP Break .....OK
  5. GO to BREAK Time Measurement
     01) Counter Test Mode (EMU 12MHz MPU 12MHz Sampling 20ns) .....OK
     02) EMU 12MHz MPU 12MHz Sampling 20ns .....OK
     03) EMU 12MHz MPU 12MHz Sampling 1.6us .....OK
     04) EMU 12MHz MPU 12MHz Sampling 52u .....OK
     05) EMU 12MHz MPU 12MHz Sampling MPU .....OK
     06) EMU 12MHz MPU 12MHz Sampling MPU/2 .....OK
     07) EMU 12MHz MPU 12MHz Sampling MPU/4 .....OK
     08) EMU 12MHz MPU 12MHz Sampling MPU/8 .....OK
     09) EMU 12MHz MPU 48MHz Sampling 20ns .....OK
```

```
     10) EMU  4MHz MPU 32MHz Sampling 20ns ..............................OK
     11) EMU 35.7MHz MPU 35.7MHz Sampling 20ns ..........................OK
     12) EMU 35.9MHz MPU 35.9MHz Sampling 20ns ..........................OK
  6. Emulation Monitor
     01) AUDRES .........................................................SKIP
     02) TRES ...........................................................SKIP
     03) ASEST ..........................................................OK
  7. G/A Break Function
     01) Address Condition ..............................................OK
     02) Data Condition .................................................OK
     03) Control Signal Condition (ASEDSHH/HL/LH/HL) .....................OK
     04) Function Code Condition (ASEST3-0) .............................OK
     05) Control Signal Condition (CB31) ................................SKIP
     06) Control Signal Condition (CB22-19) .............................OK
     07) Control Signal Condition (CB18-16) .............................OK
  8. G/A Performance Analysis Function
     01) Time Measurement (20ns Sampling) ...............................OK
     02) Task Time Measurement (20ns Sampling) ..........................OK
  9. G/A Monitor Function
     01) RUN ............................................................OK
     02) VCCDOWN ........................................................SKIP
     03) NOCLK ..........................................................OK
     04) TIMEOUT ........................................................OK
 10. G/A Parallel RAM Monitor
     01) PRAM Clear .....................................................OK
     02) PRAM Monitor  ..................................................OK
 11. G/A Trace Function
     01) Free Trace .....................................................OK
     02) Trace Stop .....................................................OK
     03) Time Stamp .....................................................OK
 12. Combination
     01) B to A Time Measurement( FPGA counter ) ........................OK
     02) B to A Time Measurement( G/A counter ) .........................OK
     03) D to C Time Measurement( G/A counter ) .........................OK
 13. Parallel Access
     01) Internal ROM Parallel Test .....................................OK
     02) Internal RAM Parallel Test .....................................OK
     03) Internal I/O Parallel Test .....................................OK
 14. H8SX/1544 Register Read/Write
     01) Register Initial Value Check ...................................OK
     02) Register Write/Verify ..........................................OK
 15. FPGA Parallel RAM Function
     01) CH0 256Byte Area Check(Byte) ...................................OK
     02) CH0 256Byte Area Check(Word) ...................................OK
     03) CH0 256Byte Area Check(Long Word) ..............................OK
     04) CH1-CH11 256Byte Area Check(Long Word) .........................OK

Normal stopped at WWW MMM DD hh:mm:ss YYYY

Tests run for xh:xmin:xs

Summary:
    Tests performed 1 time(s).
  1. Main Board Access                   : 0 Error(s)
  2. Emulation Board Access              : 0 Error(s)
  3. Evaluation Board Access             : 0 Error(s)
  4. Basic Function                      : 0 Error(s)
  5. GO to BREAK Time Measurement        : 0 Error(s)
  6. Emulation Monitor                   : 0 Error(s)
  7. G/A Break Function                  : 0 Error(s)
  8. G/A Performance Analysis Function   : 0 Error(s)
  9. G/A Monitor Function                : 0 Error(s)
 10. G/A Parallel RAM Monitor            : 0 Error(s)
 11. G/A Trace Function                  : 0 Error(s)
 12. Combination                         : 0 Error(s)
 13. Parallel Access                     : 0 Error(s)
 14. H8SX/1544 Register Read/Write       : 0 Error(s)
 15. FPGA Parallel RAM Function          : 0 Error(s)
```

Shows the time when diagnostic program execution ended.

Shows the execution time of the diagnostic program.

Shows the total number of errors for each test item.

Debugger Part

Section 1 Overview

The Debugger Part includes the following information.

Table 1.1 Debugger Part Contents

Section	Title	Content
2	Preparation before Use	This section starts with creation of a workspace and ends with connection to the emulator.
3	Debugging	This section describes this emulator ‘s peculiar debugging operation and the associated windows and dialog boxes. Refer to the High-performance Embedded Workshop user's manual about High-performance Embedded Workshop common functions as below. — Preparations for Debugging — Viewing a Program — Operating Memory — Displaying Memory Contents as Waveforms — Displaying Memory Contents as an Image — Modifying the variables — Viewing the I/O Memory — Looking at Registers — Executing Your Program — Viewing the Function Call History — Debugging with the Command Line Interface — Elf/Dwarf2 Support — Looking at Labels
4	Tutorial	This section describes how to use the emulator functions by using a tutorial program provided with the emulator.
5	Software Specifications and Notes Specific to This Product	This section describes software specifications and notes regarding the emulator.
6	Error Messages	This section describes the contents of error messages that may occur while the emulator is in use, and solutions to them.

Section 2 Preparation before Use

2.1 Method for Activating High-performance Embedded Workshop

To activate the High-performance Embedded Workshop, follow the procedure listed below.

1. Connect the emulator to the host computer.
2. Connect the user system interface cable to the connector of the emulator if you use the user system interface cable. This is not necessary when you do not use the user system interface cable.
   Turn on the emulator. Be sure to turn on the user system before supplying power to the emulator if you use the user system.
3. Activate the High-performance Embedded Workshop from [Programs] in the [Start] menu.
4. The [Welcome!] dialog box is displayed.

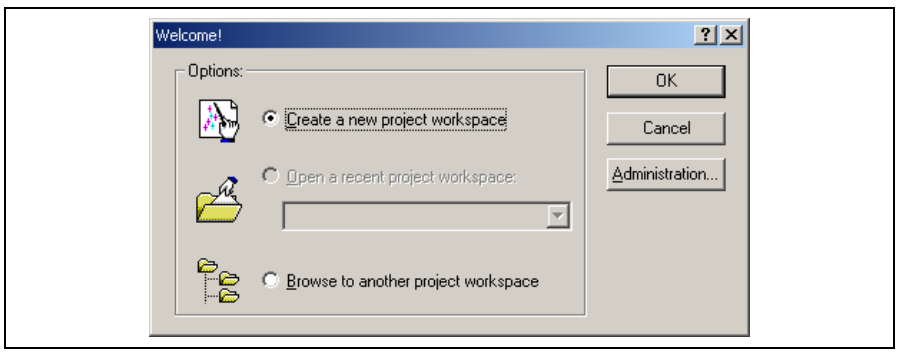


Figure 2.1 [Welcome!] Dialog Box

[Create a new project workspace] radio button: Creates a new workspace.

[Open a recent project workspace] radio button: Uses an existing workspace and displays the history of the opened workspace.

[Browse to another project workspace] radio button: Uses an existing workspace; this radio button is used when the history of the opened workspace does not remain.

In this section, we describe the following three ways to start up the High-performance Embedded Workshop:

- [Create a new project workspace] - a toolchain is not in use
- [Create a new project workspace] - a toolchain is in use
- [Browse to another project workspace]

The method to create a new workspace depends on whether a toolchain is or is not in use. Note that this emulator product does not include a toolchain. Use of a toolchain is available in an environment where the H8S, H8/300 series C/C++ compiler package or the SuperH™ RISC engine C/C++ compiler package has been installed. For details on this, refer to the manual attached to the H8S, H8/300 series C/C++ compiler package or the SuperH™ RISC engine C/C++ compiler package.

2.1.1 Creating a New Workspace (Toolchain Not Used)

1. In the [Welcome!] dialog box that is displayed when the High-performance Embedded Workshop is activated, select [Create a new project workspace] radio button and click the [OK] button.

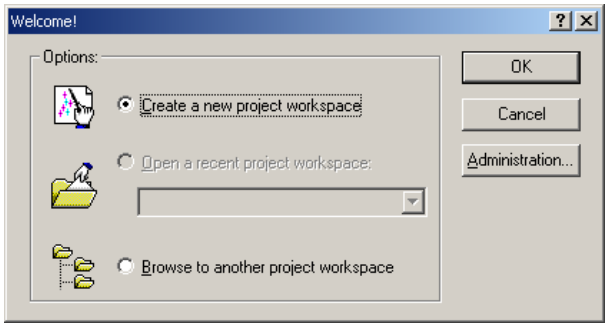


Figure 2.2 [Welcome!] Dialog Box

2. Creation of a new workspace is started. The following dialog box is displayed.

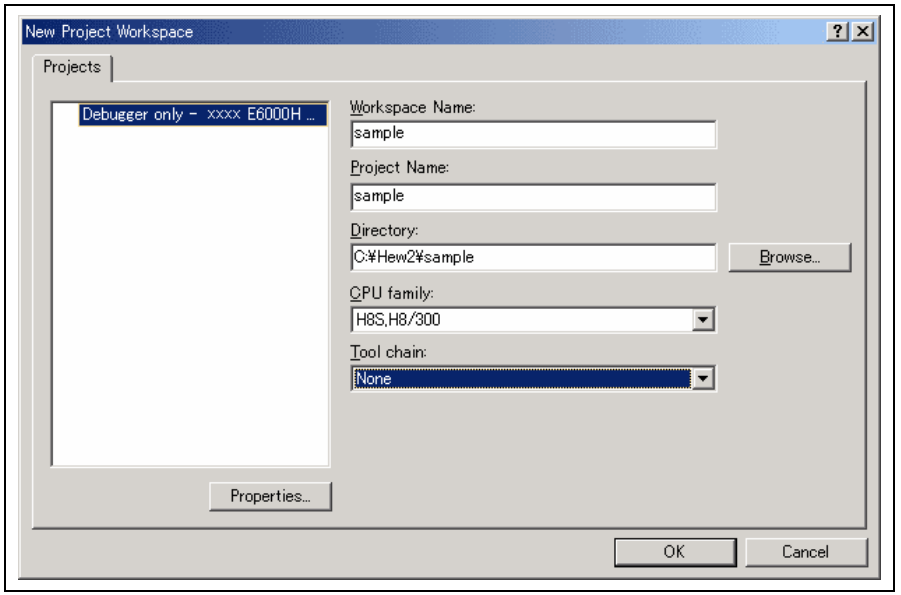


Figure 2.3 [New Project Workspace] Dialog Box

[Workspace Name] edit box: Enter the new workspace name.

[Project Name] edit box: Enter the project name. When the project name is the same as the workspace name, it needs not be entered.

[Directory] edit box: Enter the directory name in which the workspace will be created. Click the [Browse…] button to select a directory.

[CPU family] combo box: Select the target CPU family.

Other list boxes are used for setting the toolchain; the fixed information is displayed when the toolchain has not been installed.

Click the [OK] button.

3. Select the target platform of the session file. The following dialog box is displayed.

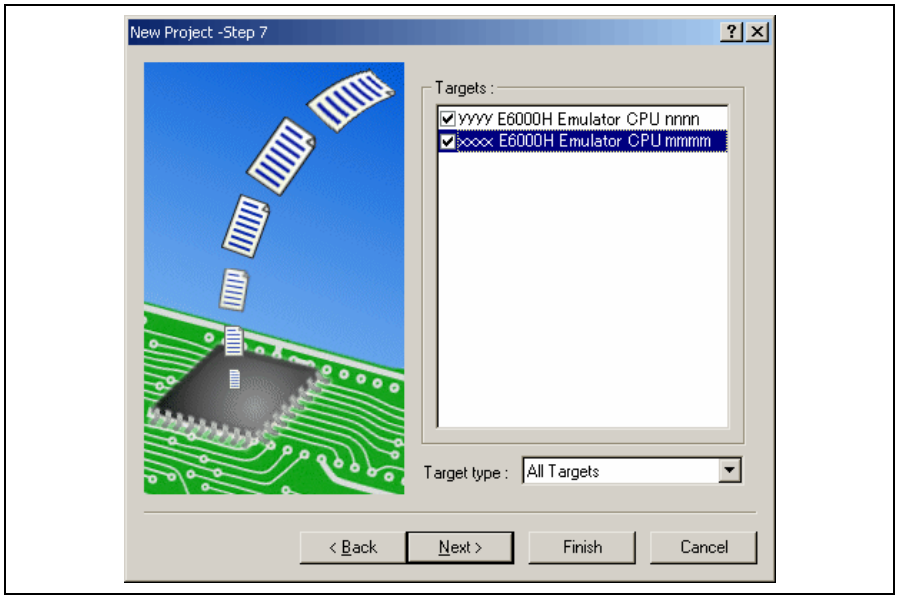


Figure 2.4 [New Project – Step 7] Dialog Box

The target platform for the session file used when the High-performance Embedded Workshop is activated must be selected here. Check the box against the target platform and then click the [Next] button.

4. Set the configuration file name. The configuration file saves the state of High-performance Embedded Workshop except for the emulator.

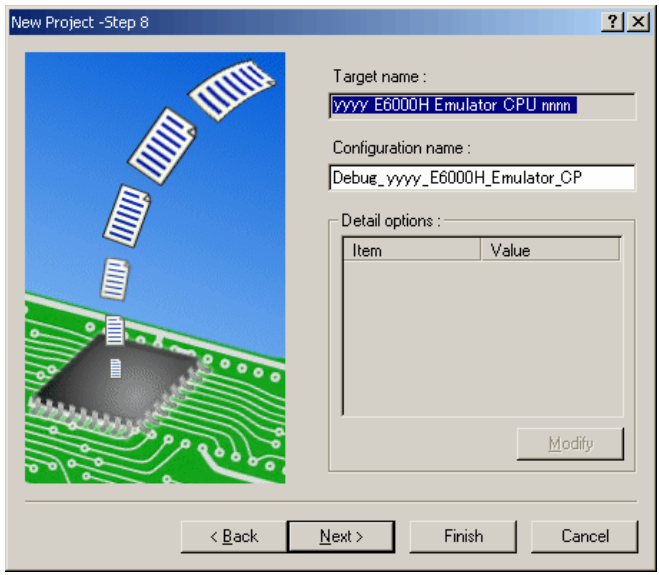


Figure 2.5 [New Project – Step 8] Dialog Box

If multiple target platforms were selected in the [New Project – Step 7] dialog box shown in figure 2.5, set the name of a configuration file for each of them, each time clicking the [Next] button to proceed to the next target.

Setting of the configuration file name is the end of the emulator settings.

Click the [Finish] button to display the [Summary] dialog box. Clicking the [OK] button activates the High-performance Embedded Workshop.

5. After the High-performance Embedded Workshop has been activated, the emulator is automatically connected. The message “Connected” is displayed on the [Debug] tab in the [Output] window to indicate the completion of connection.

2.1.2 Creating a New Workspace (Toolchain Used)

1. In the [Welcome!] dialog box that is displayed when the High-performance Embedded Workshop is activated, select the [Create a new project workspace] radio button and click the [OK] button.

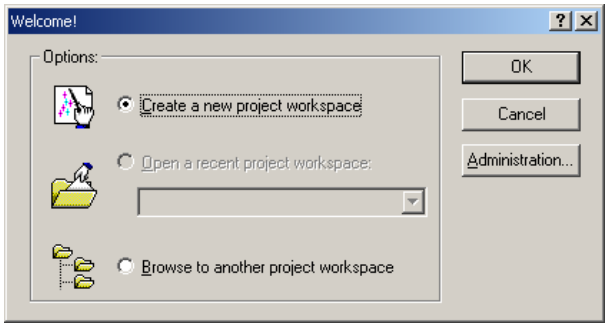


Figure 2.6 [Welcome!] Dialog Box

2. Creation of a new workspace is started. The following dialog box is displayed.

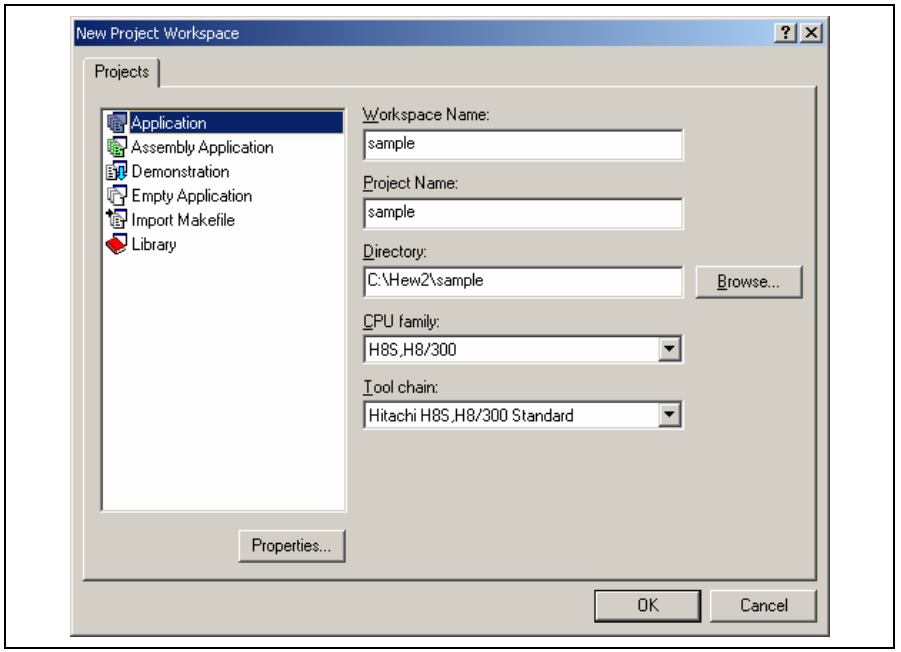


Figure 2.7 [New Project Workspace] Dialog Box

[Workspace Name] edit box: Enter the new workspace name.

[Project Name] edit box: Enter the project name. When the project name is the same as the workspace name, it needs not be entered.

[Directory] edit box: Enter the directory name in which the workspace will be created. Click the [Browse…] button to select a directory.

[CPU family] combo box: Select the target CPU family.

[Tool chain] combo box: Select the target toolchain name when using the toolchain. Otherwise, select [None].

[Project type] list box: Select the project type to be used.

Note: When [Demonstration] is selected in the emulator, note the followings:
The [Demonstration] is a program for the simulator attached to the H8S, H8/300 compiler package or the SuperH™ RISC engine C/C++ compiler package. To use the generated source file, delete the printf statement in the source file.

3. Make the required setting for the toolchain. When the setting has been completed, the following dialog box is displayed.

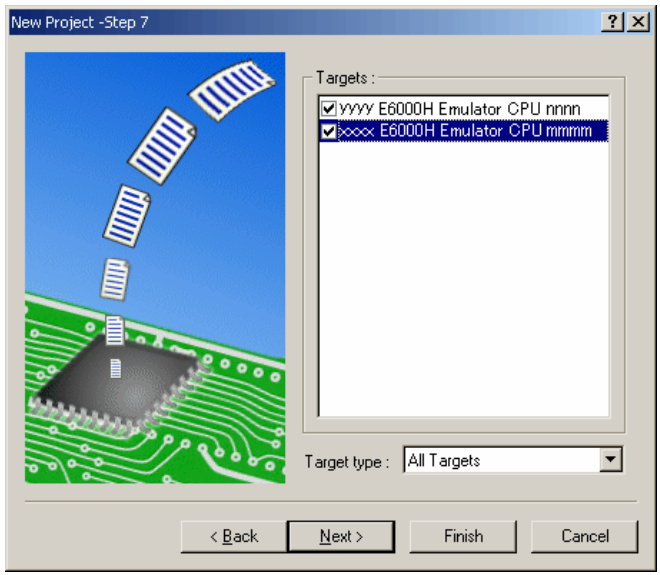


Figure 2.8 [New Project – Step 7] Dialog Box

The target platform for the session file used when the High-performance Embedded Workshop is activated must be selected here. Check the box against the target platform and then click the [Next] button.

4. Set the configuration file name. The configuration file saves the state of High-performance Embedded Workshop except for the emulator.

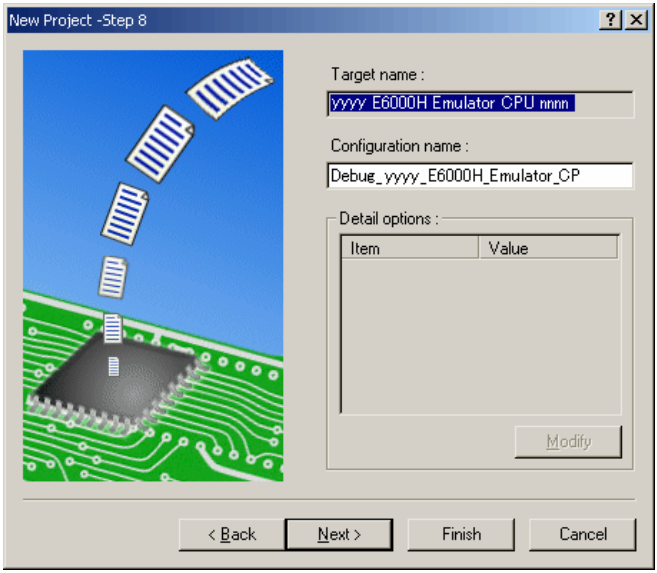


Figure 2.9 [New Project – Step 8] Dialog Box

If multiple target platforms were selected in the [New Project – Step 7] dialog box shown in figure 2.9, set the configuration file name for each of them, each time clicking the [Next] button to proceed to the next target.

Setting of the configuration file name is the end of the emulator settings.

Complete the creation of a new workspace according to the instructions on the screen. This activates the High-performance Embedded Workshop.

5. After the High-performance Embedded Workshop has been activated, connect the emulator. However, it is not necessary to connect the emulator immediately after the High-performance Embedded Workshop has been activated.

   Select either of the following two ways to connect the emulator: connecting the emulator after the setting at emulator activation or without the setting at emulator activation. For details on the connection of the emulator, refer to section 2.2, Connecting the Emulator.

2.1.3 Selecting an Existing Workspace

1. In the [Welcome!] dialog box that is displayed when the High-performance Embedded Workshop is activated, select [Browse to another project workspace] radio button and click the [OK] button.

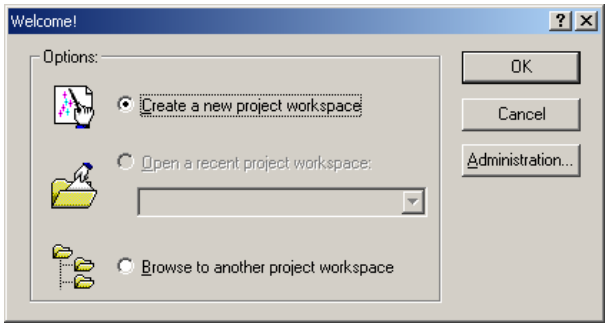


Figure 2.10 [Welcome!] Dialog Box

2. The [Open Workspace] dialog box is displayed. Select a directory in which you have created a workspace. After that, select the workspace file (.hws) and click the [Open] button.

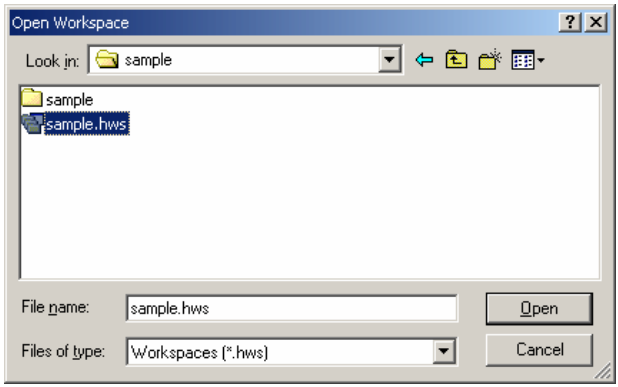


Figure 2.11 [Open Workspace] Dialog Box

3. This activates the High-performance Embedded Workshop and recovers the state of the selected workspace at the time it was saved.
   When the saved state information of the selected workspace includes connection to the emulator, the emulator will automatically be connected. To connect the emulator when the saved state information does not include connection to the emulator, refer to section 2.2, Connecting the Emulator.

2.2 Connecting the Emulator

Select either of the following two ways to connect the emulator:

(a) Connecting the emulator after the setting at emulator activation

Select [Debug -> Debug Settings…] to open the [Debug Settings] dialog box. It is possible to register the download module or the command chain that is automatically executed at activation.

When the dialog box is closed after setting the [Debug Settings] dialog box, the emulator will automatically be connected.

(b) Connecting the emulator without the setting at emulator activation

Connect the emulator by simply switching the session file to one in which the setting for the emulator use has been registered.

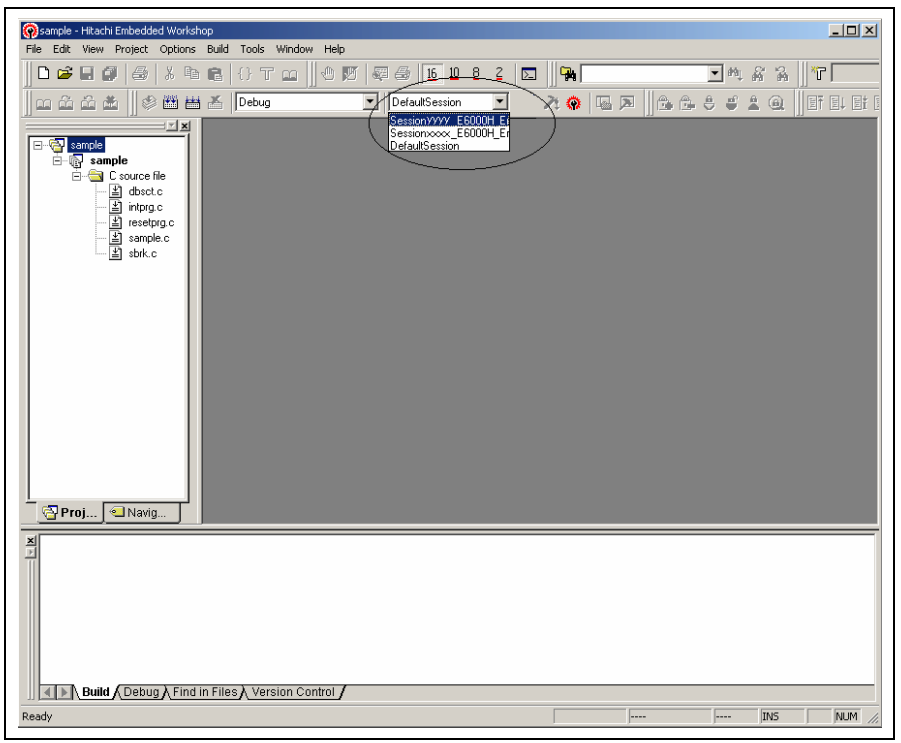


Figure 2.12 Selecting the Session File

In the list box that is circled in figure 2.12, select the session file name including the character string that has been set in the [Target name] text box in figure 2.9, [New Project – Step 8] dialog box. The setting for using the emulator has been registered in this session file.

Selecting [Debug -> Connect] connects the emulator.

2.3 Re-connecting the Emulator

When the emulator is disconnected, re-connection is possible by using the following methods.

Select [Debug -> Connect] or click the [Connect] toolbar button ( ) to re-connect the emulator.

Note: When re-connecting the emulator, the load module must be registered to the High-performance Embedded Workshop beforehand.

2.4 Ending the Emulator

The emulator can be exited by using the following two methods:

(a) Canceling the connection of the emulator being activated

Select [Debug -> Disconnect] or click on the [Disconnect] toolbar button ( ).

(b) Exiting the High-performance Embedded Workshop

1. Select [File -> Exit].
2. A message box is displayed. If necessary, click the [Yes] button to save a session. After saving a session, the High-performance Embedded Workshop exits.

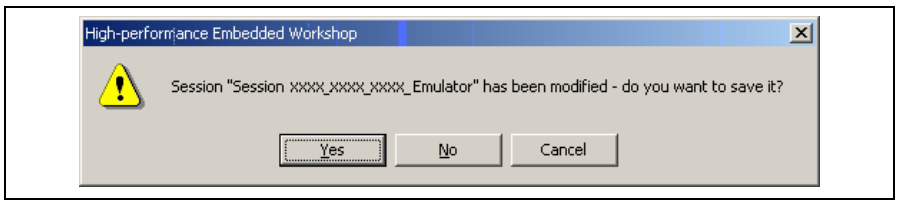

High-performance Embedded Workshop
Session "Session xxxx_xxxx_xxxx_ Emulator" has been modified - do you want to save it?
Yes
No
Cancel

Figure 2.13 [Session has been modified] Message Box

Section 3 Debugging

This section describes the debugging operations and their related windows and dialog boxes.

3.1 Setting the Environment for Emulation

The method for setting the environment for emulation is described here. This environment must be set correctly before debugging is started.

3.1.1 Opening the [Configuration Properties] Dialog Box

Selecting [Setup -> Emulator -> System…] or clicking the [Emulator System] toolbar button ( ) opens the [Configuration Properties] dialog box.

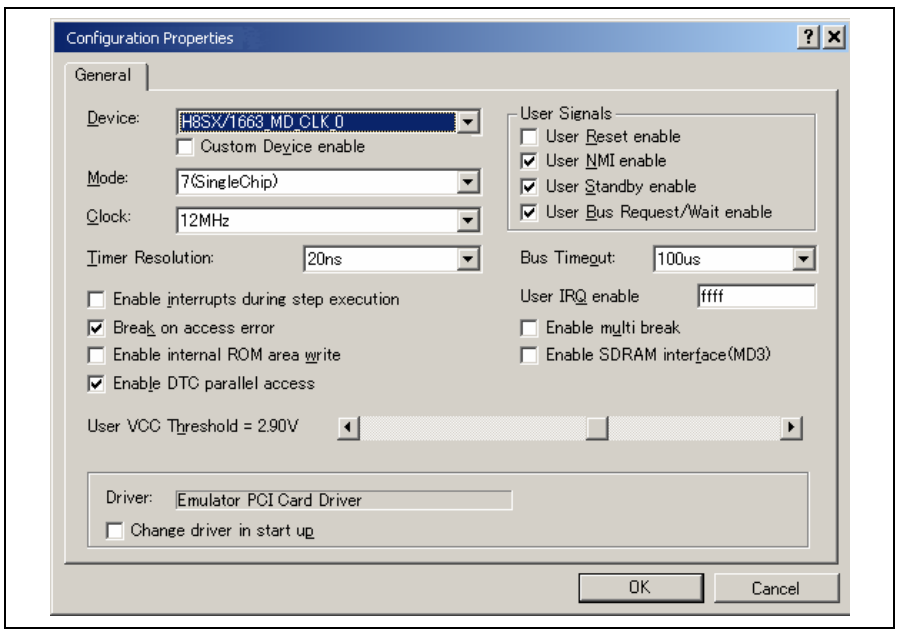


Figure 3.1 [Configuration Properties] Dialog Box ([General] Page)

[General] page

[Device]	Selects the target MCU to be emulated. See the hardware manual for details.
[Custom Device enable]	Customizes the target MCU. Checking this box allows the user to make settings on the [Custom Device] page.
[Mode]	Selects the operating mode of the target MCU.
[Clock]	Selects the speed of the input clock of the target MCU.
[Timer Resolution]	Selects the resolution of the timer for use in execution time measurement and performance analysis. Select one of the following values. Execution time measurement: 52us, 1.6us, or 20ns Performance analysis: CLOCK, CLOCK/2, CLOCK/4, or CLOCK/8
[Enable interrupts during step execution]	When this box is checked, interrupts are accepted during step execution.
[Break on access error]	When this box is checked, a break (the user program stops) occurs if your program accesses a guarded memory area or writes to a write-protected area.
[Enable internal ROM area write]	When this box is checked, writing to the on-chip ROM area is enabled. For the result of writing, see the [Status] window.
[Enable DTC parallel access]	When this box is checked, reading and writing of memory contents by parallel accesses are enabled.
[User VCC Threshold]	Sets the voltage level for the user system. [Down] will be displayed in [User VCC] of the [Extended Monitor] window when the actual user VCC of the target system is lower than the specified value.
[User Signals]	When this box is checked, the selected signal from the user system is enabled.
[Bus Timeout]	Select the bus timeout detection time. 100us, 1.6ms, 13ms, or 210ms can be selected.
[User IRQ enable]	Specify an IRQ for the trace or break condition. IRQ0 to IRQ15 are available and they can be specified by entering values in hexadecimal. Each of the bits corresponds to an IRQ. The bit location indicates the IRQ number, for example, bit D15 for IRQ15. Setting a bit to 1 allows monitoring the corresponding IRQ. If several IRQs are specified, the values will be ORed and this result will be applied.
[Enable multi break]	When this box is checked, the multibreak function is enabled. This allows breaking the program for several E6000H emulators at the same time by using a trigger input and probe pins.
[Enable SDRAM interface (MD3)]	When this box is checked, the SDRAM interface is enabled. The emulator detects the state of the MD3 pin on the user system. If the state of the MD3 pin is low, a warning message will be displayed. This option is available only when one of the following MCUs has been selected in [Device]: H8SX/1663_MD_CLK_0 H8SX/1663_MD_CLK_1 H8SX/1664_MD_CLK_0 H8SX/1664_MD_CLK_1
[Driver]	Displays the E6000H driver that is currently installed.
[Change driver in start up]	When this box is checked, selection of a driver will be available next time the emulator is connected.

3.1.2 Customizing the Settings for the Target MCU

Selecting [Custom Device enable] in the [Configuration Properties] dialog box adds the [Custom Device] page to the dialog box.

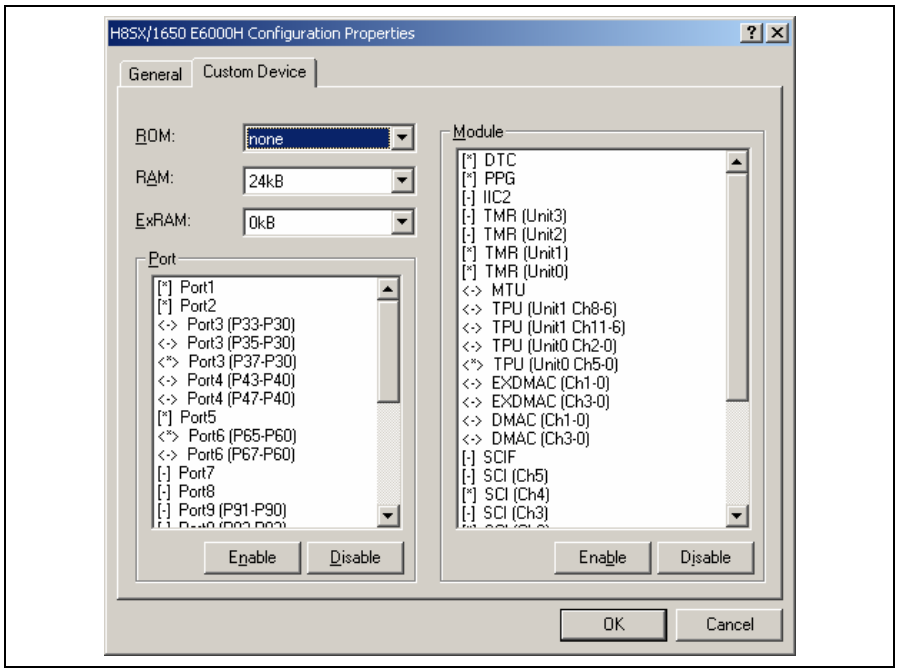


Figure 3.2 [Configuration Properties] Dialog Box ([Custom Device] Page)

Use this page to customize the target MCU. This procedure is necessary when the target MCU currently in use is not included in the list of [Device]. The items are adopted by the device last selected.

[Custom Device] page

[ROM]	Select the on-chip ROM area size.
[RAM]	Select the on-chip RAM area size.
[ExRAM]	Select the size of the expansion RAM area.
[Port]	The ports for use must be selected here. Select the ports that you want to change the settings and press the [Enable] or [Disable] button to enable or disable the ports. [*] or <*> is displayed with the enabled ports and [-] or <-> with the disabled ports.
[Modules]	The on-chip modules for use must be selected here. Select the on-chip modules that you want to change the settings and press the [Enable] or [Disable] button to enable or disable the on-chip modules. [*] or <*> is displayed with the enabled modules and [-] or <-> with the disabled modules.

3.1.3 Selecting the Interface to be Connected

Checking [Change driver in start up] on the [Configuration Properties] dialog box allows a selection of the driver next time the emulator is connected.

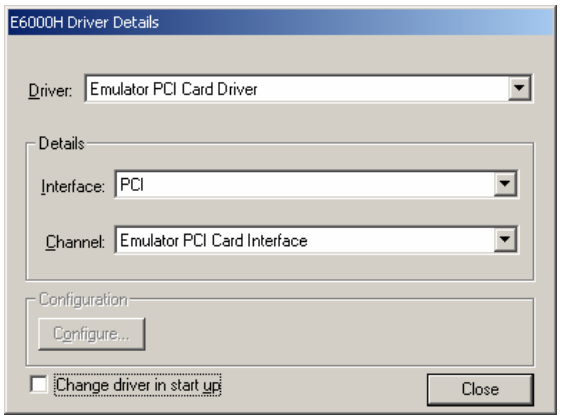


Figure 3.3 [Driver Details] Dialog Box

[Driver]: Selects the driver that connects the High-performance Embedded Workshop and the emulator.

[Details]: Sets the details of the driver being connected.

[Interface]: The name of the interface to be connected. This should not be changed in this emulator.

[Channel]: Channel for the selected interface. This should not be changed in this emulator.

[Configuration]: Driver setting.

[Configure…]: A dialog box for setting will be displayed when the driver supports the configuration dialog. Note that this item is not available with this emulator.

[Change driver in start up]:
Checking this box selects the driver when the emulator is connected the next time.

3.1.4 Opening the [Memory Mapping] Dialog Box

Selecting [Setup -> Emulator -> Memory Resource…] or clicking the [Emulator Memory Resource] toolbar button ( ) opens the [Memory Mapping] dialog box.

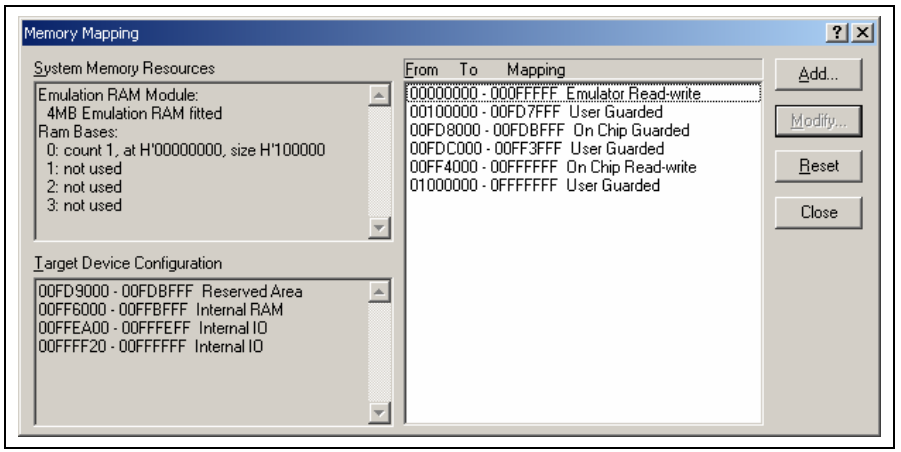


Figure 3.4 [Memory Mapping] Dialog Box

[Add...]: Displays the [Edit Memory Mapping] dialog box, allowing the user to modify the address range and attributes of a memory map.

[Modify...]: Displays the [Edit Memory Mapping] dialog box, allowing the user to modify the address range and attributes of a memory map.

[Reset]: Resets the map memory to its default settings.

[Close]: Closes the dialog box.

The memory configuration of the target MCU being emulated is displayed by the [Memory] sheet in the [Status] window.

Note: For details, refer to section 5.3, Memory Map.

3.1.5 Changing the Memory Map Setting

Clicking the [Add…] button on the [Memory Mapping] dialog box or clicking the [Modify…] button after selecting the information on the memory map setting you want to change opens the [Edit Memory Mapping] dialog box.

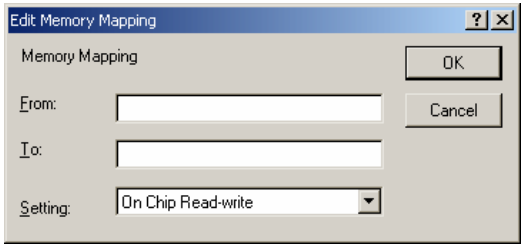


Figure 3.5 [Edit Memory Mapping] Dialog Box

Use this dialog box to change the address range and attributes of a memory map.

[From]: Enter the start address of the map range.

[To]: Enter the end address of the map range.

[Setting]: Enter the memory map setting.
The choices given are listed below. The User (external memory) and Emulator (emulation memory) attributes can be modified.

- On Chip Read-write: (Cannot be changed)
- On Chip Read-only: (Cannot be changed)
- On Chip Guarded: (Cannot be changed)
- User Read-write: Selects an external area that is readable/writable
  (Cannot be selected when the single chip mode is selected)
- User Read-only: Selects an external area that is write-protected
  (Cannot be selected when the single chip mode is selected)
- User Guarded: Selects an external area that is access-prohibited.
- Emulator Read-write: Sets readable/writable emulation memory as an external area.
- Emulator Read-only: Sets write-protected emulation memory as an external area.
- Emulator Guarded: Sets access-prohibited emulation memory as an external area.
- No-access area: Space that includes no memory

Emulation memory can be used as replacement of a ROM or flash memory on the user system. When the user program accesses to an access-prohibited area or writes to a write-protected area, a break occurs due to the illegal access. The user can set the [Configuration Properties] dialog box so that an illegal access will cause a break. Accesses to memory with the debugging function in use, as in the [Memory] window, are available regardless of the attribute that has been selected.

Note: For details, refer to section 5.3, Memory Map.

3.2 Downloading a Program

This section describes how to download a program and view it as source code or assembly-language mnemonics.

Note: After a break has occurred, the [Editor] window displays the location of the present program counter (PC). In most cases, for example if an Elf/Dwarf2-based project is moved from its original path, the source file may not be automatically found. In this case, a source file browser dialog box is displayed to allow you to manually locate the source file.

3.2.1 Downloading a Program

A load module to be debugged must be downloaded.

To download a program, select the load module from [Debug -> Download] or select [Download] from the popup menu opened by clicking the right-hand mouse button on the load module in [Download modules] of the [Workspace] window.

Note: Before downloading a program, it must be registered to the High-performance Embedded Workshop as a load module.

3.2.2 Viewing the Source Code

To view a source file's code, double-click on its icon in the file tree, or right-click on the source file and select the [Open] option on the pop-up menu. The [Editor] window is displayed.

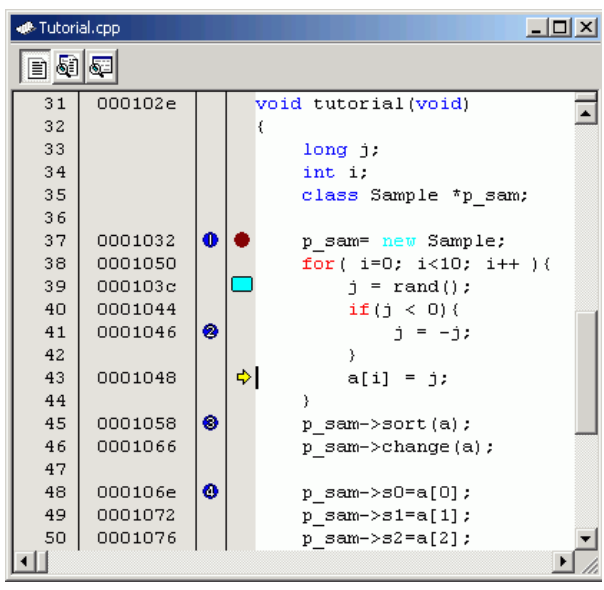


Figure 3.6 Editor Window

In this window, the following items are shown on the left as information on lines.

- 1st column (Line Number column): A line number for the source file
- 2nd column (Source Address column): Address information for the source line
- 3rd column (On Chip Break column): On-chip breaks
- 4th column (S/W Breakpoints column): PC, bookmark, and breakpoint information

The text area is displayed in the right part of the [Editor] window.

Line Number column

This column displays the line number for the source file.

Source Address column

When a program is downloaded, an address for the current source file is displayed on the Source address column. These addresses are helpful when setting the PC value or a breakpoint.

On Chip Break column

The On Chip Break column displays the following items:

: On-chip break channel 1

: On-chip break channel 2

: On-chip break channel 3

: On-chip break channel 4

These are also set by using the popup menu.

S/W Breakpoints column

This column displays the following items:

: A bookmark is set.

: A PC breakpoint is set.

: PC location

➲ To switch off a column in all source files

1. Click the right-hand mouse button on the [Editor] window or select the [Edit] menu.
2. Click the [Define Column Format…] menu item.
3. The [Global Editor Column States] dialog box is displayed.
4. A check box indicates whether the column is enabled or not. If it is checked, the column is enabled. If the check box is gray, the column is enabled in some files and disabled in others. Deselect the check box of a column you want to switch off.
5. Click the [OK] button for the new column settings to take effect.

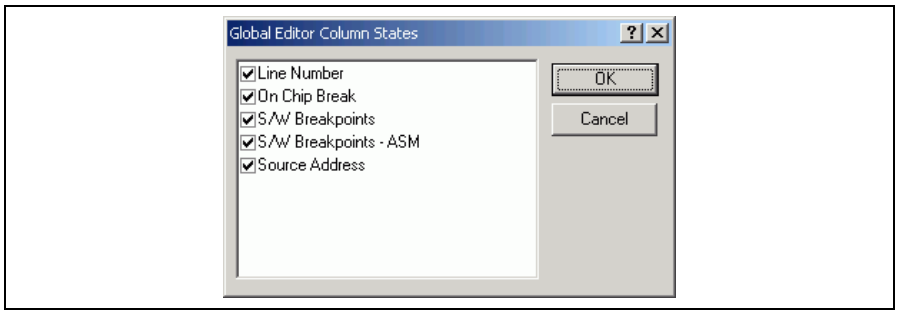


Figure 3.7 [Global Editor Column States] Dialog Box

➲ To switch off a column in one source file

1. Open the source file which contains the column you want to remove and click the [Edit] menu.
2. Click the [Columns] menu item to display a cascaded menu item. The columns are displayed in this popup menu. If a column is enabled, it has a tick mark next to its name. Clicking the entry will toggle whether the column is displayed or not.

3.2.3 Viewing the Assembly-Language Code

If you have a source file open, right-click to open the pop-up menu and select the [View Disassembly] option to open a Disassembly view at the same address as the current Source view.

It is also possible to view the disassembly using the new integrated [Disassembly view] in the source file.

If you do not have a source file, but wish to view code at assembly-language level, then select one of the following operations:

Click on the View Disassembly toolbar button ( ).

Choose the [View -> Disassembly…] menu option.

Press Ctrl + D.

The [Disassembly] window opens at the current PC location.

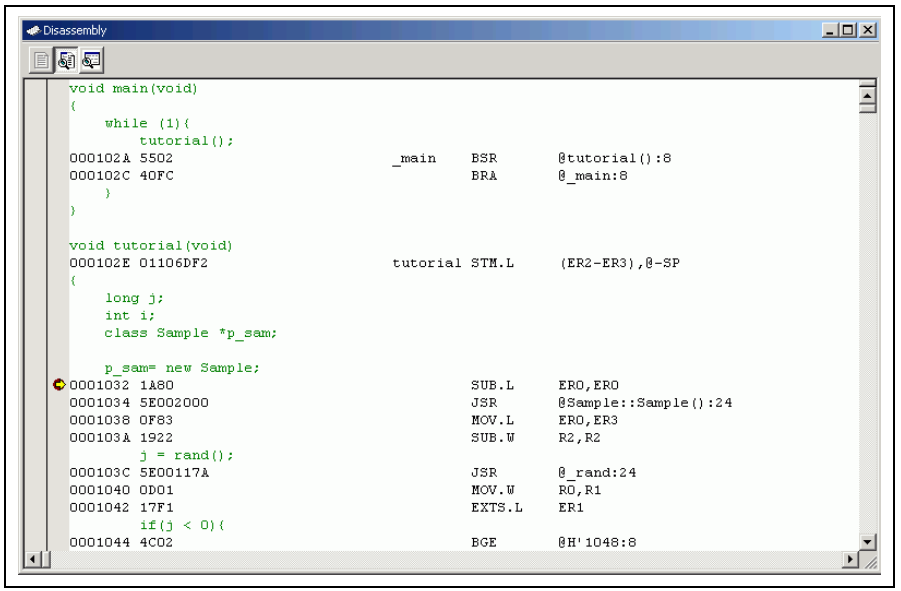


Figure 3.8 [Disassembly] Window

In this window, the following information is shown on the left as information lines.

- First column (On-chip break column): On-chip breaks
- Second column (S/W Breakpoints - ASM column): PC and breakpoint information

This window is used in the same way as the source code window.

3.2.4 Modifying the Assembly-Language Code

You can modify the assembly-language code by double-clicking on the instruction that you want to change. The [Assembler] dialog box will be opened.

Figure 3.9 [Assembler] Dialog Box

The address, instruction code, and mnemonic are displayed. Enter a new instruction or edit the old instruction in the [Mnemonics] field. Pressing the [Enter] key will replace the memory content with the new instruction and move on to the next instruction. Clicking the [OK] button will replace the memory content with the new instruction and close the dialog box. Clicking the [Cancel] button or pressing the [Esc] key will close the dialog box without modifying the memory contents.

Note: The assembly-language code being displayed is the current memory content. If the memory contents are changed the [Assembler] dialog box and the [Disassembly] window will show the new assembly-language code, but the source file displayed in the [Editor] window will be unchanged. This is the same even if the source file contains assembly codes.

3.2.5 Viewing a Specific Address

When you are viewing your program in the [Disassembly] window, you may wish to look at another area of your program's code. Rather than scrolling through a lot of code in the program, you can go directly to a specific address. Select [Set Address…] from the popup menu, and the dialog box shown in figure 3.10 is displayed.

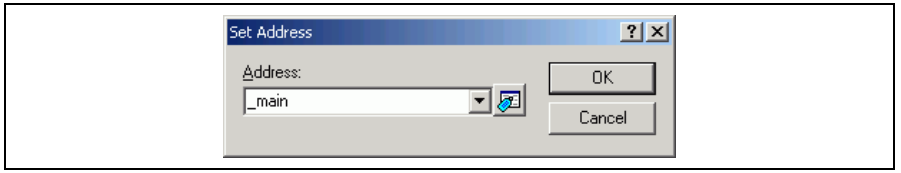


Figure 3.10 [Set Address] Dialog Box

Enter the address in the [Address] edit box and either click on the [OK] button or press the Enter key. A label name can also be specified as the address. When an overloaded function or a class name is entered, the [Select Function] dialog box opens for you to select a function.

3.2.6 Viewing the Current Program Counter Address

Wherever you can enter an address or value into the High-performance Embedded Workshop, you can also enter an expression. If you enter a register name prefixed by the hash character, the contents of that register will be used as the value in the expression. Therefore, if you enter the expression #pc in the [Set Address] dialog box, the [Editor] or [Disassembly] window will display the current PC address. It allows the offset of the current PC to be displayed by entering an expression with the PC register plus an offset, e.g., #PC+0x100.

3.3 Viewing the Current Status

Choose [View -> CPU -> Status] or click the [View Status] toolbar button ( ) to open the [Status] window and see the current status of the debugging platform.

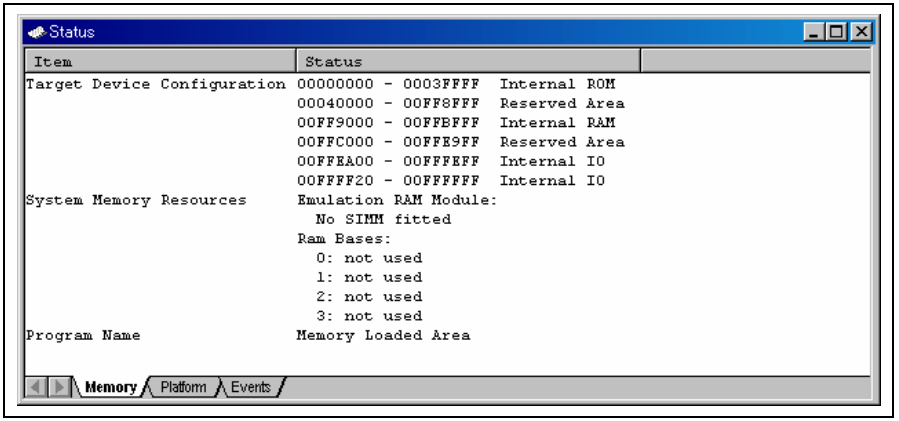


Figure 3.11 [Status] Window

The [Status] window has following three sheets:

- [Memory] sheet
  Displays information about the current memory status including the memory mapping resources and the areas used by the currently loaded object file.
- [Platform] sheet
  Displays information about the environment for emulation, typically including CPU type and emulation mode.
- [Events] sheet
  Displays information about the current event (breakpoint) status, including resource information.

Note: The items that can be set in this window depend on the emulator in use. For details, refer to the online help.

3.4 Reading and Displaying the Emulator Information Regularly

Use the [Extended Monitor] window to know the changing information on the emulator no matter the user program is running or halted.

Note: The extended monitor function does not affect the execution of the user program since it monitors the user system or the signal output from the target MCU in the emulator by using the emulator's hardware circuit.

3.4.1 Opening the [Extended Monitor] Window

Selecting [View -> CPU -> Extended Monitor] or clicking the [Extended Monitor] toolbar button ( ) displays this window. The interval of updating the display is approximately 1,000 ms during user program execution or 5,000 ms while breaking, respectively.

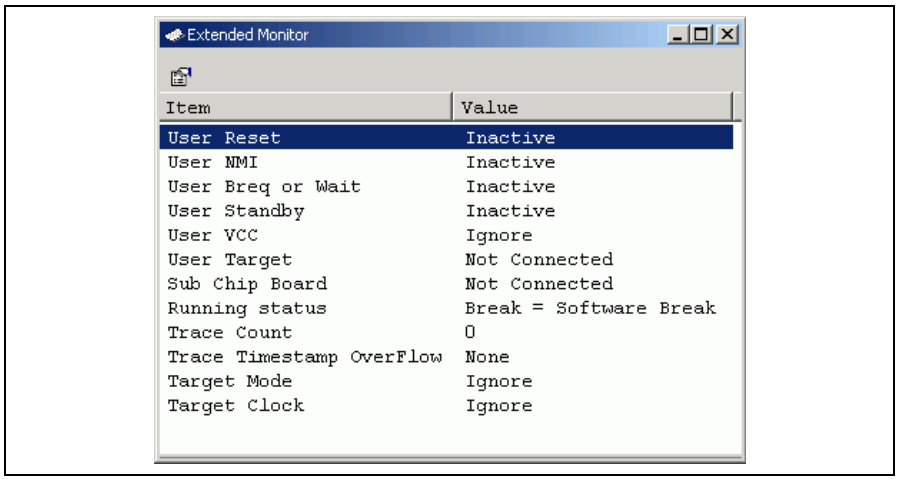


Figure 3.12 [Extended Monitor] Window

3.4.2 Selecting Items to be Displayed

Selecting [Properties…] from the popup menu displays the [Extended Monitor Configuration] dialog box.

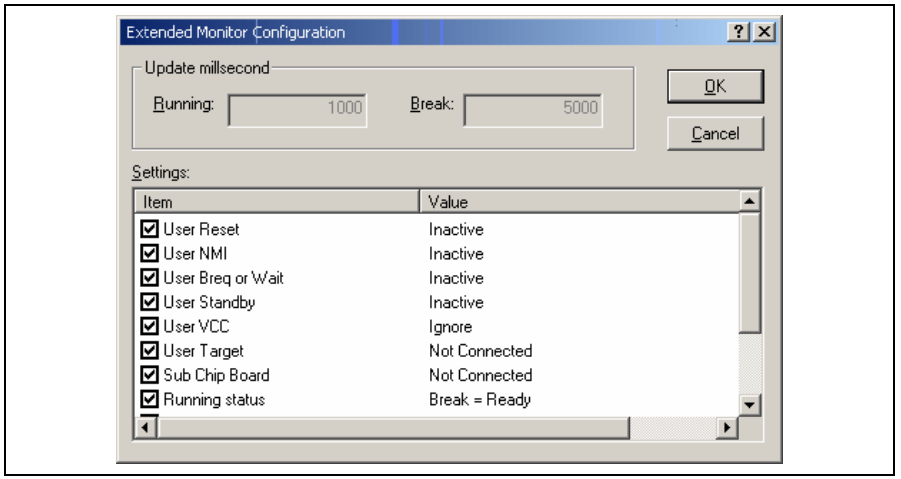


Figure 3.13 [Extended Monitor Configuration] Dialog Box

This dialog box allows the user to set the items to be displayed in the [Extended Monitor] window.

Note: The items that can be set in this window depend on the emulator in use. For details, refer to the online help.

3.5 Displaying Memory Contents in Realtime

Use the [Monitor] window to monitor the memory contents during user program execution. In the Monitor function, the realtime operation is retained since the bus monitoring circuit of the emulator sets the read/write signal of the MCU as a trigger and holds the address bus and data bus values to update the displayed contents of the memory.

Up to eight points or 256 bytes in total can be set by using the eight monitoring channels on the bus monitoring circuit. It is possible that a part or all of monitoring ranges is overlapped.

Notes: 1. Monitoring is impossible for an area, such as an on-chip timer counter, where no internal write signal is generated to update a value.
2. The procedure to display or modify the contents of memory differs depending on the product. If the display of memory contents is updated during execution of the user program, realtime emulation may not be available. For details, refer to section 5.4, Displaying and Modifying the Contents of Memory.

3.5.1 Opening the [Monitor] Window

To open the [Monitor] window, select [View -> CPU -> Monitor -> Monitor Setting...] or click the [Monitor] toolbar button ( ) to display the [Monitor Settings] dialog box.

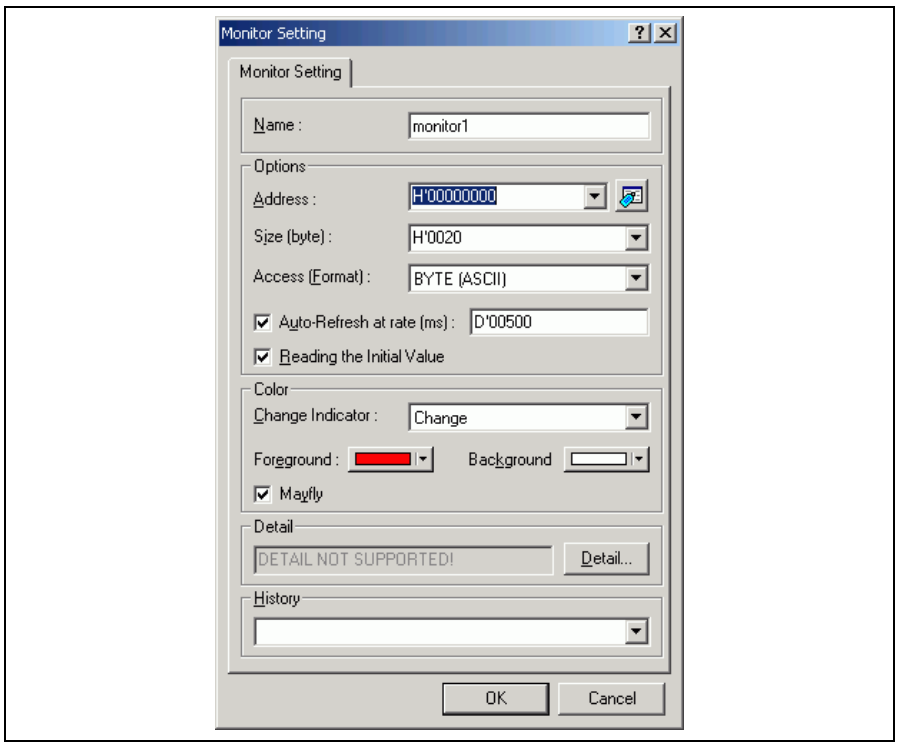


Figure 3.14 [Monitor Setting] Dialog Box

[Name]: Decides the name of the monitor window.

[Options]: Sets monitor conditions.

- [Address]: Sets the start address for monitoring.
- [Size]: Sets the range for monitoring.
- [Access]: Sets the access size to be displayed in the monitor window.
- [Auto-Refresh at rate]: Sets the interval for acquisition by monitoring (500 ms at minimum).
- [Reading the Initial Value]: Selects reading of the values in the monitored area when the monitor window is opened.

[Color]: Sets the method to update monitoring and the attribute of colors.

- [Change Indicator]: Selects how to display the values that have changed during monitoring (available when [Reading the Initial Value] has been selected).

  No change: No color change.

  Change: Color is changed according to the [Foreground] and [Background] options.

  Gray: Those data with values that have not been changed are displayed in gray.

  Appear: A value is only displayed after changed.
- [Foreground]: Sets the color used for display (available when [Change] has been selected).
- [Background]: Sets the background color (available when [Change] has been selected).
- [Mayfly]: A check in this box selects restoration of the color of those data which have not been updated in a specified interval to the color selected in the [Background] option. The specified interval is the interval for monitor acquisition (available when [Change], [Gray], or [Appear] has been selected).

[Detail]: Sets the items specific to the emulator.

[History]: Displays the previous settings.

Notes: 1. In this emulator, odd addressees cannot be specified as the start addresses for monitoring.
2. Selection of the foreground or background color may not be available depending on the operating system in use.

After setting, clicking the [OK] button displays the [Monitor] window.

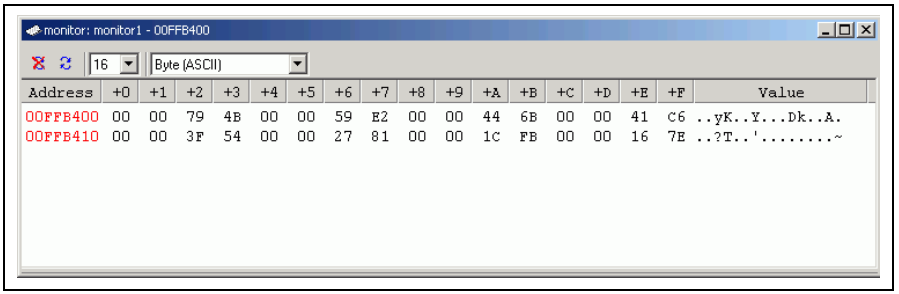


Figure 3.15 [Monitor] Window

During user program execution, the display is updated according to the setting value of the auto-update interval.

Note: Select [Refresh] from the popup menu when data is not displayed correctly after changing the address or content of memory.

3.5.2 Changing the Monitor Settings

Selecting [Monitor Setting…] from the popup menu of the [Monitor] window displays the [Monitor Setting] dialog box, which allows the settings to be changed.

Colors, the size of accesses, and the display format can be easily changed from [Color] or [Access] of the popup menu.

3.5.3 Temporarily Stopping Update of the Monitor

During user program execution, the display of the [Monitor] window is automatically updated according to the auto-update interval. Select [Lock Refresh] from the popup menu of the [Monitor] window to stop the update of display. The characters in the address section are displayed in black, and the update of display is stopped.

Selecting [Lock Refresh] again from the popup menu cancels the stopped state.

3.5.4 Deleting the Monitor Settings

Selecting [Close] from the popup menu of the [Monitor] window to be deleted closes the [Monitor] window and deletes the monitor settings.

3.5.5 Monitoring Variables

Using the [Watch] window refers to the value of any variables.

When the address of the variable registered in the [Watch] window exists within the monitoring range that has been set by the Monitor function, the value of the variable can be updated and displayed.

This function allows checking the content of a variable without affecting the realtime operation.

For the [Watch] window, refer to section 3.6.3, [Watch] Window.

3.5.6 Hiding the [Monitor] Window

When using the Monitor function to monitor the value of a variable from the [Watch] window, hide the [Monitor] window for the effective use of the screen.

The current monitoring information is listed as the submenu when selecting [Display -> CPU -> Monitor]. The list consists of the [Monitor] window name and the address to start monitoring.

When the left of the list is checked, the [Monitor] window is being displayed.

Selecting items of the [Monitor] window you want to hide from the monitor setting list displays no [Monitor] window and removes the check mark at the left of the list.

To display the [Monitor] window again, select the hidden the [Monitor] window.

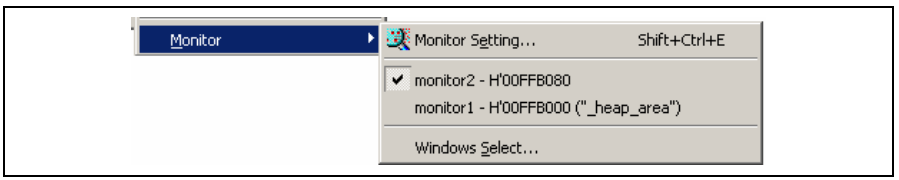


Figure 3.16 Monitor Setting List

3.5.7 Managing the [Monitor] Window

Selecting [Display -> CPU -> Monitor -> Windows Select…] displays the [Windows Select] dialog box. In this window, the current monitoring condition is checked and the new monitoring condition is added, edited, and deleted in succession.

Selecting multiple monitoring conditions enables a temporary stop of update, hiding, and deletion.

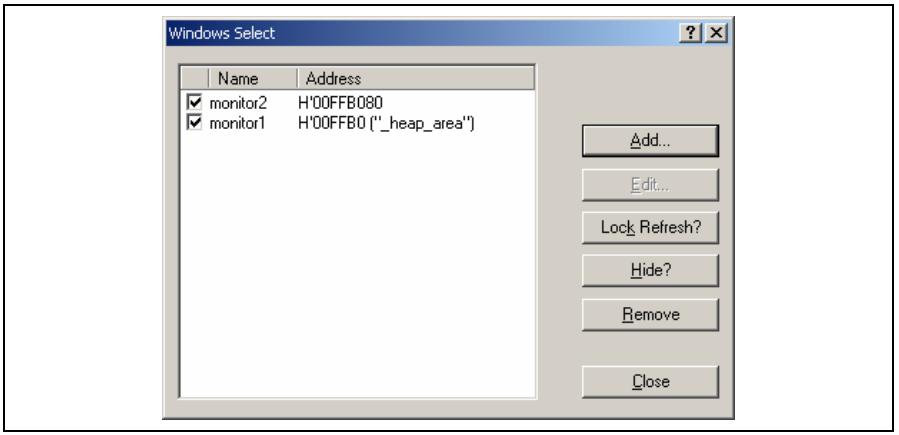


Figure 3.17 [Windows Select] Dialog Box

[Add]: Adds a new monitoring condition.

[Edit]: Changes the settings of the selected [Monitor] window (disabled when selecting multiple items).

[Lock Refresh/Unlock Refresh]:
Automatically updates or stops updating the display of the selected [Monitor] window.

[Hide/UnHide]: Displays or hides the selected [Monitor] window.

[Remove]: Removes the selected monitoring conditions.

[Close]: Closes this dialog box.

3.6 Looking at Variables

This section describes how you can look at variables in the source program.

3.6.1 [Watch] Window

You can view any value in the [Watch] window.

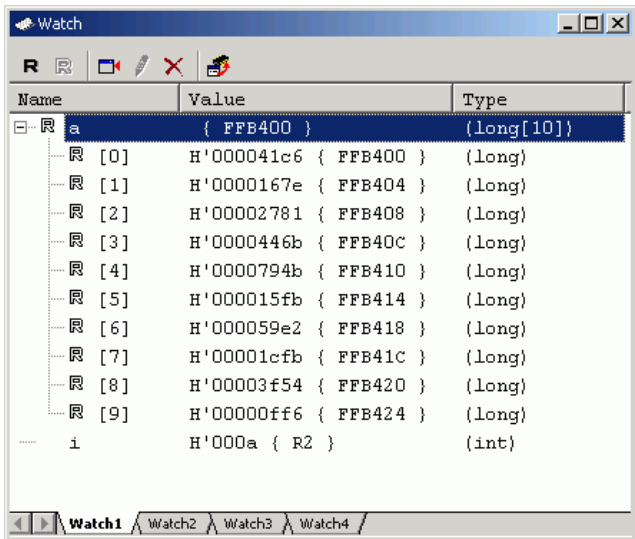

Watch
Name
Value
Type
a { FFB400 } (long[10])
[0] H'000041c6 { FFB400 } (long)
[1] H'0000167e { FFB404 } (long)
[2] H'00002781 { FFB408 } (long)
[3] H'0000446b { FFB40C } (long)
[4] H'0000794b { FFB410 } (long)
[5] H'000015fb { FFB414 } (long)
[6] H'000059e2 { FFB418 } (long)
[7] H'00001cfb { FFB41C } (long)
[8] H'00003f54 { FFB420 } (long)
[9] H'00000ff6 { FFB424 } (long)
i H'000a { R2 } (int)
Watch1
Watch2
Watch3
Watch4

Figure 3.18 [Watch] Window

The [R] mark shows that the value of the variable can be updated during user program execution.

It is possible to recognize the method for updating the value during user program execution according to the color of the [R] mark.

Blue-outline [R]: The address of the variable is within the range that has been set for the monitoring function and the data is readable by using the monitoring function.

Blue [R]: An updated value of the data at this location has been read by the monitoring function.

Black-outline [R]: The address of the variable is outside the range that has been set for the monitoring function and the data is not readable by using the monitoring function.

Black [R]: A value has been updated by reading the normal data.

Notes: 1. This function can be set per variable or per element or body for structures of data.
2. The color of an [R] in the [Name] column changes according to the monitoring settings.
3. A variable that is allocated to a register cannot be selected for monitoring.
4. The procedure to display or modify the contents of memory differs depending on the product. If the display of memory contents is updated during execution of the user program, realtime emulation may not be available. For details, refer to section 5.4, Displaying and Modifying the Contents of Memory.
5. The H8SX/1527 E6000H, H8SX/1527R E6000H and H8SX/1544 E6000H emulators incorporate a dedicated on-chip RAM monitor of 256 bytes × 12 points for the watch function that only allows the contents of the on-chip RAM to be displayed in realtime. Enabling Auto Update after registering variables automatically allows use of this on-chip RAM monitor, while disabling Auto Update or deleting variables cancels use of the on-chip RAM monitor. When Auto Update is enabled while all of the 12 points of this on-chip RAM monitor are in use, the values will be updated by reading data as usual. To display the contents of memory other than the on-chip RAM in realtime, use the normal monitor.

3.7 Using the Event Points

The emulator has the event point function to support breakpoints of the following three types.

Software breakpoints: Execution of the user program stops when the instruction at the specified address is fetched. Up to 255 software breakpoints can be set. Any content at the specified address is replaced by a break instruction (a dedicated instruction for use with the emulator). The software breakpoint cannot be set in the write-protected area such as ROM area or flash memory area on the user system. The user can set a software breakpoint in the [Editor] or [Disassembly] window.

On-chip breakpoints: These break functions built in the MCU. Conditions on the address bus, data bus, bus/area, and satisfaction count can be set. The on-chip breakpoint can be set even in the ROM area or flash memory area on the user system. It is also possible to set a sequential breakpoint consisted of several on-chip breakpoints. The user can set an on-chip breakpoint in the [Editor] or [Disassembly] window.

On-emulator breakpoints: On-emulator break functions are implemented by dedicated hardware in the E6000H station. Conditions on the address bus, data bus, bus/area, external probe signals, external interrupt signals, and satisfaction count can be set. The on-emulator breakpoint can be set even in the ROM area or flash memory area on the user system. As the emulator hardware provides this function, several cycles may be required until a break occurs after satisfaction of a condition.

Software, on-chip, and on-emulator breakpoints can be set in the [Event] window.

Select [View -> Code -> Eventpoints] or click the [Eventpoints] toolbar button ( ) to open the [Event] window.

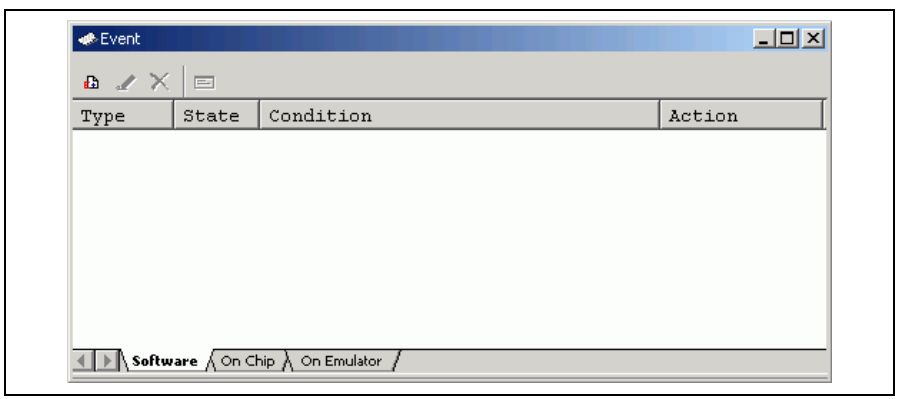


Figure 3.19 [Event] Window

The [Event] window has the following three sheets:

[Software] sheet: Displays the settings made for software breakpoints. It is also possible to set, modify, and cancel software breakpoints.

[On Chip] sheet: Displays or sets on-chip breakpoints.

[On Emulator] sheet: Displays or sets on-emulator breakpoints.

Note: For notes on event points, refer o section 5.6, Event Functions.

3.7.1 Setting a Software Breakpoint

Use the [Software] sheet on the [Event] window to display, change, or add settings for software breakpoints.

Select [Add...] or [Edit…] from the popup menu displayed on the [Software] sheet. The [Breakpoint Properties] dialog box (the [Software Break] page) will appear.

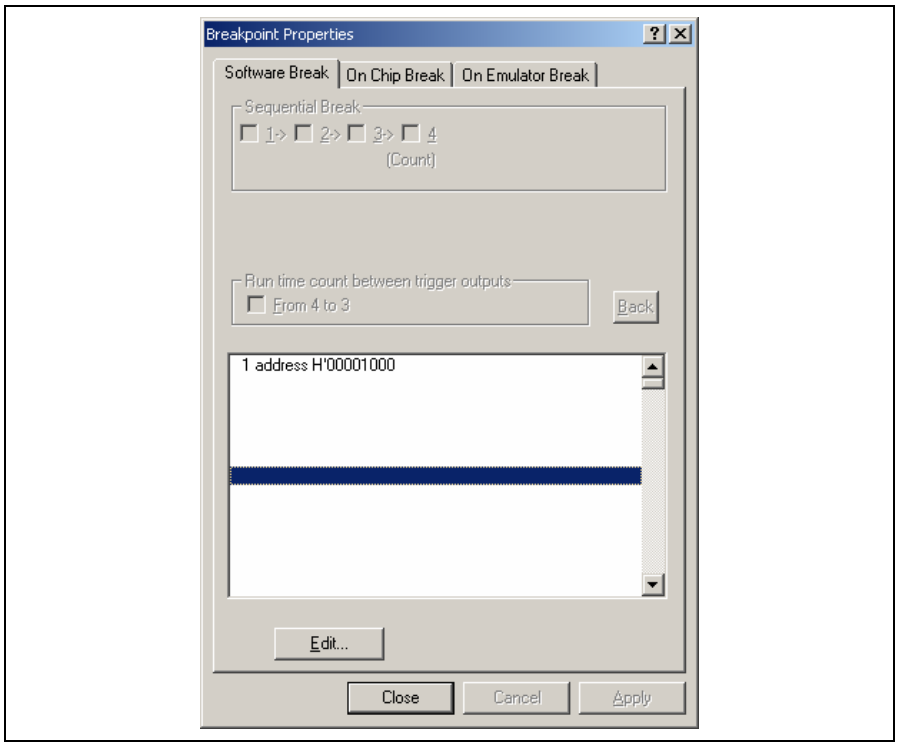


Figure 3.20 [Breakpoint Properties] Dialog Box ([Software Break] Page)

To add a new software breakpoint, select an empty line from the list box on the [Software Break] page and click the [Edit…] button. To change existing settings, select the software breakpoint that you want to change from the list box and click the [Edit…] button. The [Software Break] dialog box is displayed.

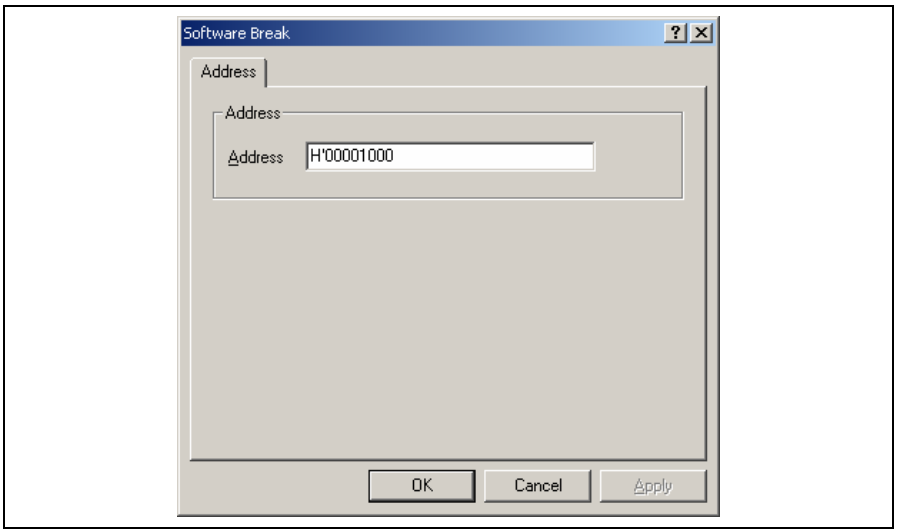


Figure 3.21 [Software] Dialog Box ([Address] Page)

Specify the breakpoint's address in the [Address] edit box and click the [OK] button.

3.7.2 Setting an On-Chip Breakpoint

Use the [On Chip] sheet on the [Event] window to display, change, or add settings for on-chip breakpoints.

Select [Add...] or [Edit…] from the popup menu displayed on the [On Chip] sheet. The [Breakpoint Properties] dialog box (the [On Chip Break] page) will appear.

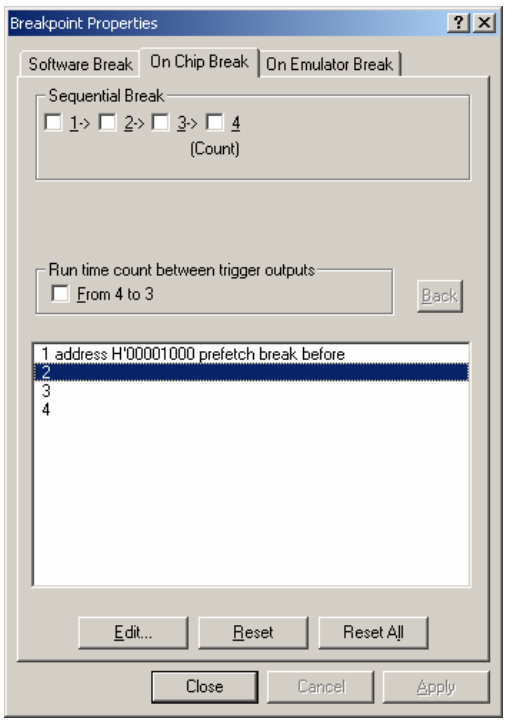


Figure 3.22 [Breakpoint Properties] Dialog Box ([On Chip Break] Page)

[Sequential Break]: Use this item to specify a sequential break consisted of on-chip breakpoints. A sequential break allows breaking the program when the conditions of several channels are satisfied sequentially. Two to four points can be selected for this function. The conditions are satisfied from 1 to 4, in order. Check a box to use the sequential break function (select 3 to use two points, 2 to use three points, and 1 to use four points).

[Run time count between trigger outputs]:
Measures the time between two points by using channels 3 and 4. After channel 4 has been satisfied, the time is measured when channel 3 is satisfied. The result is displayed in [RunTime Count] on the [Platform] sheet of the [Status] window.

[Back]: Puts the setting back to the previous state at the time the dialog box has been displayed.

List box: Displays the current settings for each of the channels. If no setting has been made for a channel, only the channel number is displayed here. When a channel is used for the sequential break function, S is displayed next to the channel number.

[Edit…]: Clicking this button opens the [On Chip Break Channel n] dialog box (n: channel number), which allows the user to set a break condition for a selected channel.

[Reset]: Clears the settings made for the selected channel.

[Reset All]: Clears the settings made for all of the channels.

The user can set more complex break conditions in the [On Chip Break Channel n] dialog box by a combination of conditions provided on pages [Address], [Data], [Bus/Area], [Count], and [Action].

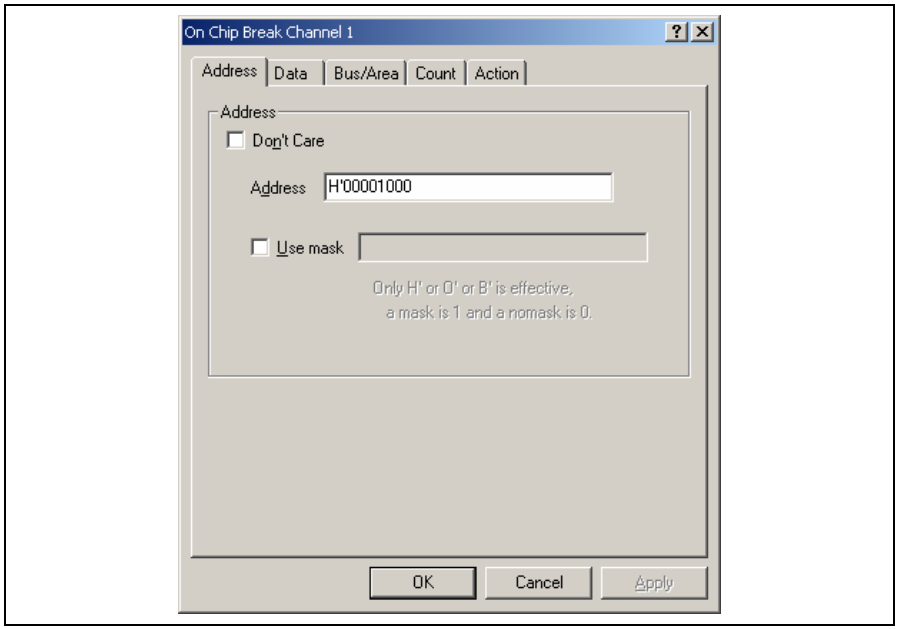


Figure 3.23 [On Chip Break Channel n] Dialog Box

[Address]: Sets address bus conditions.

[Don't Care]: Selects no address bus condition.

[Address]: Sets an address bus value.

[Use mask]: Sets mask conditions. Set the mask bits if [Use mask] is selected. Masked bits satisfy this break condition regardless of their values.

[Data]: Sets data bus conditions.

[Don't Care]: Selects no data bus condition.

[Value]: Sets a data bus value.

[Use mask]: Sets mask conditions. Set the mask bits if [Use mask] is selected. Masked bits satisfy this break condition regardless of their values.

[Access Size]: Selects the data-access size.

[Bus/Area]: Sets access type, bus status, and read/write cycle conditions.

[Access type]: Sets access type conditions.

[Bus State]: Sets bus status conditions. When [Don't Care] has been selected, no bus status condition can be set.

[Read/Write]: Sets read/write conditions. When [Don't Care] has been selected, no read/write condition can be set.

[Count]: Sets the satisfaction count of the condition. When [Don't Care] has been selected, the satisfaction count is defined as 1.

[Action]

[Break]: Halts execution when the selected condition has been satisfied.

[After execution]: Halts execution after the address at which the condition has been satisfied.

[Before execution]: Halts execution before the address at which the condition is satisfied.

3.7.3 Settings an On-Emulator Breakpoint

Use the [On Emulator] sheet on the [Event] window to display, change, or add settings for on-emulator breakpoints.

Select [Add...] or [Edit…] from the popup menu displayed on the [On Chip] sheet. The [Breakpoint Properties] dialog box (the [On Emulator Break] page) will appear.

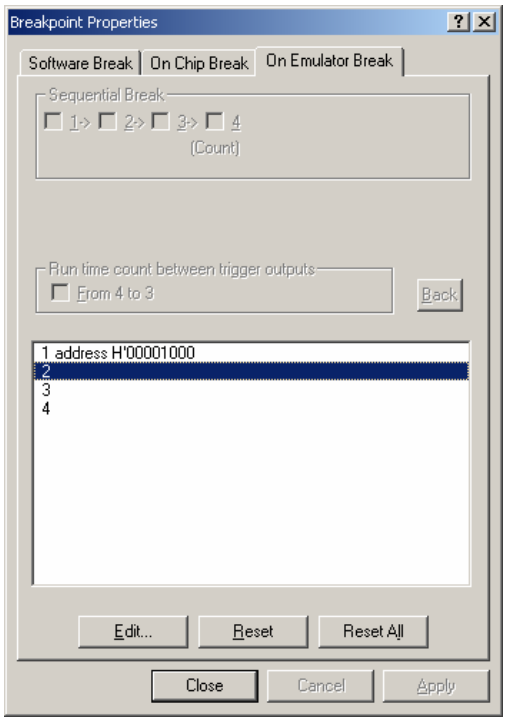


Figure 3.24 [Breakpoint Properties] Dialog Box ([On Emulator Break] Page)

List box: Displays the current settings for each of the channels. If no setting has been made for a channel, only the channel number is displayed here.

[Edit…]: Clicking this button opens the [On Emulator Break Channel n] dialog box (n: channel number), which allows the user to set a break condition for a selected channel.

[Reset]: Clears the settings made for the selected channel.

[Reset All]: Clears the settings made for all of the channels.

The user can set more complex break conditions in the [On Emulator Break Channel n] dialog box by a combination of conditions provided on pages [Address], [Data], [Bus/Area], [Probe], [Interrupt], and [Count].

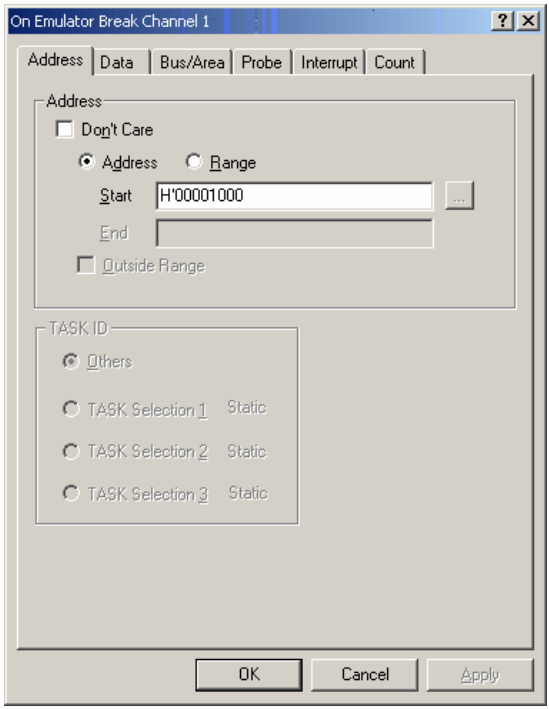


Figure 3.25 [On Emulator Break Channel n] Dialog Box

[Address]: Sets address conditions.

[Don't Care]: Selects no address bus condition.

[Address]: Select this button to set the address bus value specified in [Start] as the break condition.

[Range]: A break occurs in the range of the address bus values specified from [Start] (start address) to [End] (end address).

[Outside Range]: Select this option to generate a break with an address bus outside the range set in [Range].

[f()…]: The address range of a function can be set by [Start] and [End]. For details, refer to section 5.12, Input Format.

[Data]: Sets data conditions.

[Don't Care]: Selects no data bus condition.

[Value]: Sets a data bus value.

[Use mask]: Sets mask conditions. Set the mask bits if [Use mask] is selected. Masked bits satisfy this break condition regardless of their values.

[Except this value]: Sets a value other than that has been specified as the data bus condition.

[Access Size]: Selects the data-access size.

[Position]: Sets a data bus value as a number. The position of the valid data bus is specified.

[Long]: None

[Word]: 4n: Upper word
4n + 2: Lower word

[Byte]: 4n: Upper byte of the upper word
4n + 1: Lower byte of the upper word
4n + 2: Upper byte of the lower word
4n + 3: Lower byte of the lower word

[Bus/Area]: Sets access type, bus status, and read/write cycle conditions.

[Access type]: Sets access type conditions. When [Don't Care] has been selected, no access type condition can be set.

[Bus State]: Sets bus status conditions. When [Don't Care] has been selected, no bus status condition can be set.

[Area]: Sets area conditions. When [Don't Care] has been selected, no area condition can be set.

[On-chip ROM]: Selects the on-chip ROM area as the condition.

[On-chip RAM]: Selects the on-chip RAM area as the condition.

[On-chip I/O]: Selects the on-chip I/O area as the condition.

[External 32bit]: Selects an external area with the 32-bit width where no emulation memory is allocated as the condition.

[External 16bit]: Selects an external area with the16-bit width where no emulation memory is allocated as the condition.

[External 8bit]: Selects an external area with the 8-bit width where no emulation memory is allocated as the condition.

[Emulator 32bit]: Selects an external area with the 32-bit width where emulation memory is allocated as the condition.

[Emulator 16bit]: Selects an external area with the 16-bit width where emulation memory is allocated as the condition.

[Emulator 8bit]: Selects an external area with the 8-bit width where emulation memory is allocated as the condition.

[Read/Write]: Sets read/write conditions. When [Don't Care] has been selected, no read/write condition can be set.

[Probe]: Sets the levels (high or low) of the external probe signals (PRB1 to PRB4) as the condition. When [Don't Care] has been selected, the level of the selected probe signal cannot be set as the condition.

[Interrupt]: Sets the levels (high or low) of the IRQ signals as the condition. When [Don't Care] has been selected, the level of the IRQ signal cannot be set as the condition.

[Count]: Sets a satisfaction count condition. When [Don't Care] has been selected, the satisfaction count is defined as 1.

3.7.4 Editing Event Points

Handlings for settings other than software breakpoints, on-chip breakpoints, and on-emulator breakpoints are common.

3.7.5 Modifying Event Points

Select an event point to be modified, and choose [Edit...] from the popup menu to open the dialog box that corresponds the event, which allows the user to modify the event conditions. The [Edit...] menu is only available when one event point is selected.

3.7.6 Enabling an Event Point

Select an event point and choose [Enable] from the popup menu to enable the selected event point.

3.7.7 Disabling an Event Point

Select an event point and choose [Disable] from the popup menu to disable the selected event point. When an event point is disabled, the event point will remain in the list, but an event will not occur when the specified conditions have been satisfied.

3.7.8 Deleting an Event Point

Select an event point and choose [Delete] from the popup menu to remove the selected event point. To retain the event point but not have it cause an event when its conditions are met, use the [Disable] option (see section 3.15.7, Disabling an Event Point).

3.7.9 Deleting All Event Points

Choose [Delete All] from the popup menu to remove all event points.

3.7.10 Viewing the Source Line for an Event Point

Select an event point and choose [Go to Source] from the popup menu to open the [Editor] or [Disassembly] window at the address of the event point. The [Go to Source] menu is only available when one event point that has the corresponding source file is selected.

3.7.11 Setting a Data Condition for a Variable in the Source Program

Open the [Editor] window that shows the variable for which you want to set a data condition. Place the mouse cursor on the variable name and select [E6000H On Chip Break] from the popup menu. The on-chip break conditions currently set are displayed on the channels. Select a channel on which you want to set a data condition. After selecting the channel, the [On Chip Break] page of the [Breakpoint Properties] dialog box opens, while the data condition for the variable is set.

3.8 Viewing the Trace Information

The emulator acquires the results of each instruction execution into the trace buffer as trace information and displays it in the [Trace] window. The conditions for the trace information acquisition can be specified in the [Trace Acquisition] dialog box.

Since trace information in bus-cycles is acquired by the hardware circuit and stored in the trace buffer, the realtime operation is retained. The [Trace] window displays the content of the trace buffer, which records up to 128-k bus cycles from the last program run and is always updated.

Note: For notes on the trace functions, refer to section 5.7, Trace Functions.

3.8.1 Opening the [Trace] Window

To open the [Trace] window, choose [View -> Code -> Trace] or click the [Trace] toolbar button ( ).

3.8.2 Acquiring Trace Information

When the emulator does not set the acquisition condition of the trace information, all bus cycles are acquired by default without any condition (free trace mode).

In the free trace mode, trace acquisition is started with the execution of the user program and stopped by halting the user program. The acquired trace information is displayed in the [Trace] window.

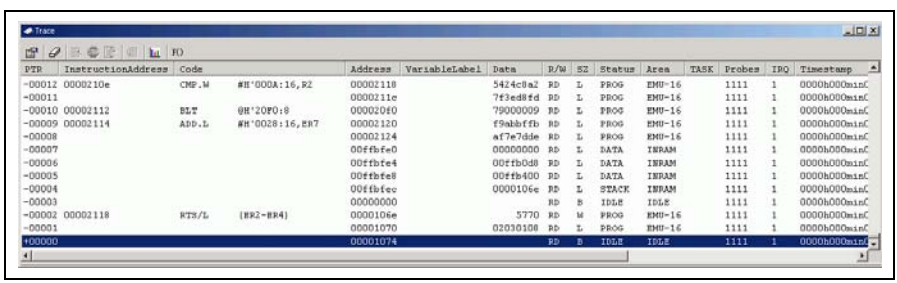


Figure 3.26 [Trace] Window

This window displays the following trace information items:

[PTR]: Cycle number in the trace buffer. When the most recent record is record 0, earlier record numbers go backwards (-1, -2, ...). If a delay count has been set, the cycle number where the trace stop condition has been satisfied is record 0. For the cycle (during delay) executed until the trace has stopped, earlier record numbers go forward (+1, +2, ...) the most recent record.

[Instruction Address]: Executed instruction address (8-digit hexadecimal)

[Code]: Instruction code of the address

[Address]: Address on the processor bus

[Data]: Data in byte, word, or longword units, displayed as 2-digit, 4-digit, or 8-digit hexadecimal

[Variable Label]: Label information of the address (if defined) when Data (CPU data access cycle) is selected in [Status].

[R/W]:	Whether access was read (RD) or write (WR)
[SZ]:	Selects the size of an access as B (byte), W (word), or L (longword).
[Status]:	Bus status during this cycle; DTC operation, PROG (prefetch), Data (CPU data access cycle), DMAC (DMAC cycle), or STACK (STACK cycle).
[Area]:	Memory area being accessed; ROM (on-chip ROM), RAM (on-chip RAM), I/O (on-chip I/O), 8- 16-, or 32-bit EXT (external), or 8- 16-, or 32-bit EMU (emulation memory).
[Probes]:	A 4-bit binary number showing the four probe pins in the order of Probe 4, Probe 3, Probe 2, and Probe 1 from the left.
[IRQ]:	Status of IRQ inputs
[Timestamp]:	Time stamp of the record. Time stamps start from zero each time the user program is executed. The minimum unit used in time measurement depends on the time stamp clock rate selected in the [Trace Acquisition] dialog box.
[Timestamp-Difference]	Difference from the timestamp value shown on the previous line
[Source]:	Source program of the executed instruction address
[Label]:	Label information of the address (if defined)

It is possible to hide any column not necessary in the [Trace] window. Selecting a column you want to hide from the popup menu displayed by clicking the right-hand mouse button on the header column hides that column. To display the hidden column, select the column from the said popup menu again.

3.8.3 Specifying Trace Acquisition Conditions

The capacity of the trace buffer is limited. When the buffer becomes full, the oldest trace information is overwritten. Setting the trace acquisition condition allows acquisition of useful trace information and effective use of the trace buffer.

There are the following types of trace acquisition conditions.

Free trace: Acquires trace information continuously from the start of the user program execution to the occurrence of a break (only when no trace acquisition condition is set).

Sequential trace stop: Specifies the order of trace acquisition conditions to be satisfied and stops trace acquisition when all of the conditions are satisfied. It is possible to set up to seven pass points and one reset point. No break will occur even when the trace acquisition stops.

Trace stop due to trace buffer overflow: Stops trace acquisition when the trace buffer in the emulator station overflows. No break will occur even when the trace acquisition stops.

Trace stop: Stops trace acquisition when the specified conditions are satisfied. In this mode, trace acquisition stops without stopping the user program execution. Up to 12 points can be set independently as trace stop conditions. No break will occur even when the trace acquisition stops.

Address range trace: Acquires trace information of instructions or operands accessed in the range (subroutine) between the start and end addresses. Note that, however, when the selected subroutine calls another subroutine, no trace information will be acquired from the called subroutine. Up to 12 points can be set independently as the address ranges.

Conditional trace: Only acquires trace information from the points where the specified conditions are satisfied. Up to 12 points can be set independently as the conditions.

Address range conditional trace: Accesses instructions or operands in the range (subroutine) between the start and end addresses and only acquires trace information in the bus cycles that satisfy the conditions. This mode is a combination of address range trace and conditional trace. Up to six points can be set independently as the address ranges with conditions.

Point to Point trace: Acquires trace information from the satisfaction of the address condition set as a start condition to that of the address condition set as an end condition.

Execution time measurement: Measures execution time between two points by using the trace acquisition conditions.

Trigger output: Outputs a pulse from trigger pins when the specified conditions are satisfied.

The trace acquisition condition is set in the [Trace Acquisition] dialog box that is displayed by selecting [Acquisition…] from the popup menu.

The [Trace Acquisition Properties] dialog box has the pages [Condition] and [Other].

(1) [Condition] page

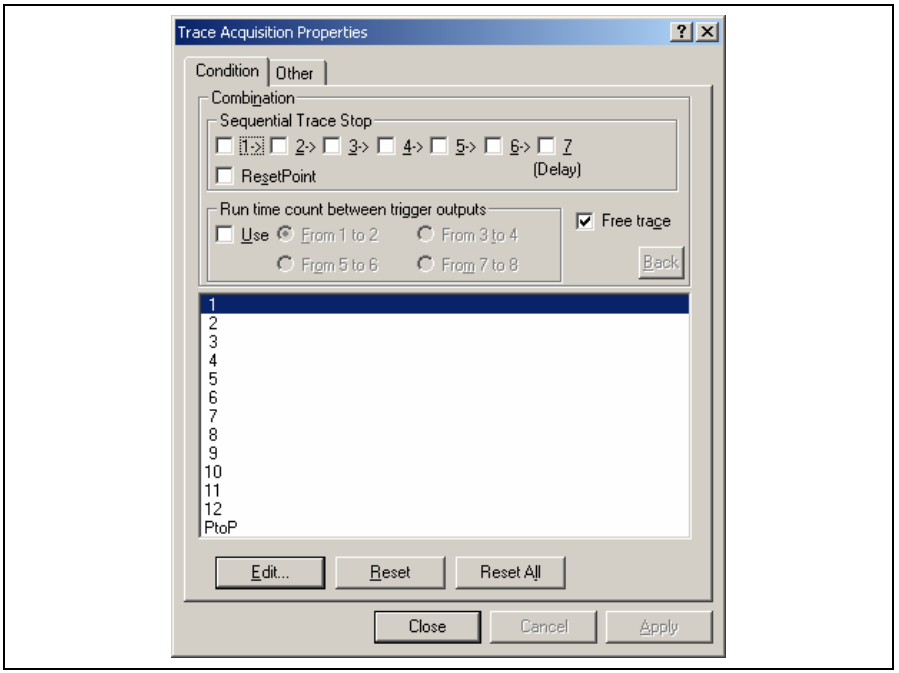


Figure 3.27 [Trace Acquisition Properties] Dialog Box ([Condition] Page)

[Sequential Trace Stop]: Use this option to set a sequential trace stop by using channels 1 to 7. The sequential trace stop function allows trace acquisition to stop when the conditions of several channels are satisfied in the specified order. Two to seven pass points and one reset point are selectable as sequential trace stop conditions. The conditions are satisfied in the order of 1 to 7. To use a sequential trace stop, select the checkbox of the channel. To set a reset condition, select the [Reset Point] checkbox. Channel 8 is used for a reset condition. When a reset condition is satisfied, all the sequential trace stop conditions that have been satisfied will be cleared and the emulator starts checking the first condition again. When a sequential trace stop is enabled, no setting is available for the channels (out of 1 to 7) that are not used for the sequential trace stop function.

[Run time count between trigger outputs]:
Selects channels for use in execution time measurement. Clicking [Use] allows measurement of time in tracing. There are four types of channel combinations consisted of those for the start and the end of measurement: 1-2, 3-4, 5-6, and 7-8. Execution time from satisfaction of the condition on the start channel to that of the condition on the end channel is measured. Any trigger output from other channels is invalid because this measurement uses a trigger output.

[Free trace]: Selects the free trace mode. When [Free trace] is enabled, any trace acquisition condition set will be ignored.

[Back]:	Puts the setting back to the previous state at the time the dialog box has been displayed.
List box:	Displays the current settings for each of the channels. If no setting has been made for a channel, only the channel number is displayed here. When a channel is used for the sequential trace stop function, S is displayed next to the channel number. When a reset condition for a sequential trace stop is enabled, R is displayed next to channel 8. PtoP is for use in the Point to Point trace. UNUSED is displayed next to the channel number if that channel is not available.
[Edit…]:	Clicking this button opens the [Trace Acquisition Condition Channel n] dialog box (n: channel number or PtoP), which allows the user to set a break condition for a selected channel.
[Reset]:	Clears the settings made for the selected channel.
[Reset All]:	Clears the settings made for all of the channels.

(2) [Other] page

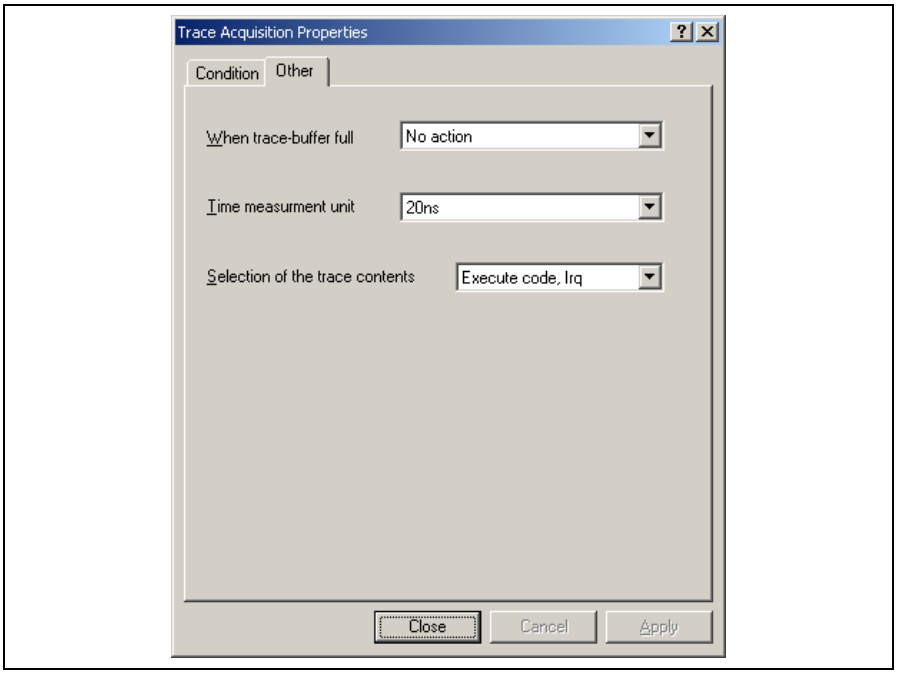


Figure 3.28 [Trace Acquisition] Dialog Box ([Other] Page)

[When trace-buffer full]: Selects an action to take when the trace buffer becomes full.

[No action]: Overwrites the oldest information in the trace buffer.

[Stop trace]: Stops trace acquisition without stopping the user program execution.

[Stop execution and trace]: Stops the user program execution.

[Time measurement unit]: Selects the minimum time unit for the time stamping of the bus trace information.

[52us]: Time stamping is in minimum time units of 52 µs.

[1.6us]: Time stamping is in minimum time units of 1.6 µs.

[20ns]: Time stamping is in minimum time units of 20 ns.

[Clock]: Time stamping is in terms of the number of bus-clock cycles, i.e., is synchronized with the cycles of the system clock signal (ϕ).

[Clock/2]: Time stamping is in terms of the number of bus-clock cycles, i.e., is synchronized with 1/2 cycle of the system clock signal (ϕ).

[Clock/4]: Time stamping is in terms of the number of bus-clock cycles, i.e., is synchronized with 1/4 cycle of the system clock signal (ϕ).

[Clock/8]: Time stamping is in terms of the number of bus-clock cycles, i.e., is synchronized with 1/8 cycle of the system clock signal (ϕ).

[Selection of the trace contents]:

Selects whether or not to display time stamps, IRQs, or executed instructions.

[Detailed time stamp]: Acquires 32-bit time stamp information. No IRQ or executed instruction is displayed.

[Execute code,Irq]: Displays IRQs and executed instructions. Lower 16 bits of time stamps are fixed to 0.

(3) [Trace Acquisition Condition Channel n] dialog box

Use this dialog box to set pass points and a reset point for a sequential trace stop, and conditions in the address range trace, address range conditional trace, conditional trace, Point to Point trace, execution time measurement, and a trigger output.

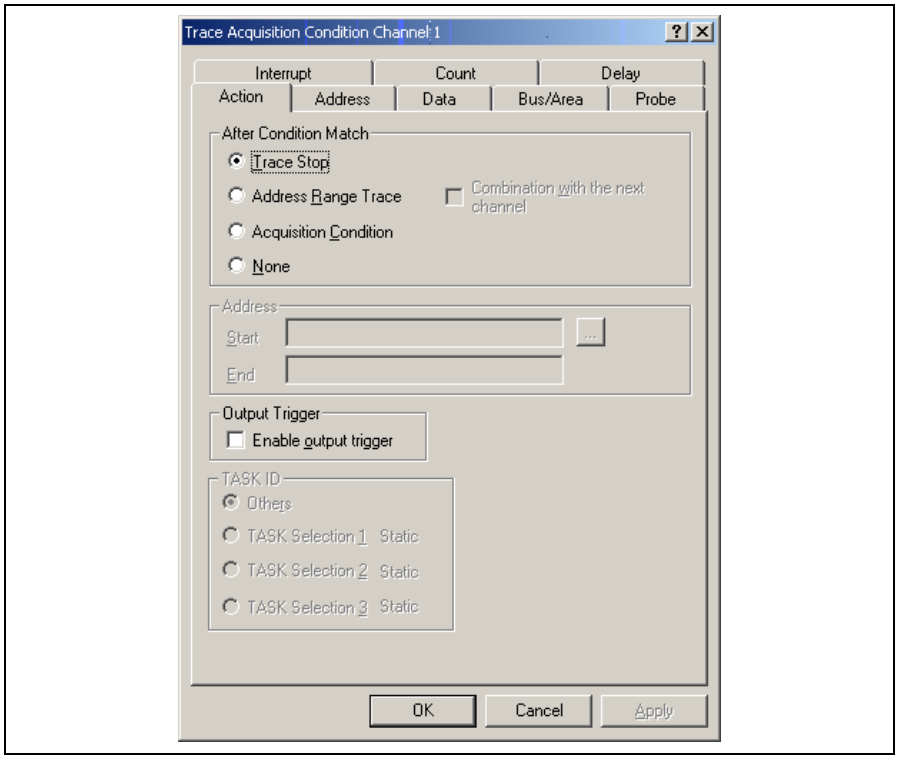


Figure 3.29 [Trace Acquisition Condition Channel n] Dialog Box

The [Trace Acquisition Condition Channel n] dialog box has pages [Action], [Address], [Data], [Bus/Area], [Probe], [Interrupt], [Count], and [Delay]. The user can make more complex settings by a combination of conditions provided on these pages.

[Action]

[After Condition Match]: Selects an action to take when a condition is satisfied.

[Trace Stop]: Selects a trace stop.

[Address Range Trace]: Selects an address range trace. Selecting this option and checking [Combination with the next channel] allows an address range conditional trace.

[Acquisition Condition]: Selects a conditional trace.

[None]: Select this option if you do not want to take any of the actions listed above. This is useful for a trigger output or execution time measurement.

[Address]: Sets the start and end addresses of the range in the address range trace, address range conditional trace, or Point to Point trace.

[Start]: Set the start address.

[End]: Set the end address.

[f()…]: The address range of a function can be set by [Start] and [End].

[Output Trigger]: Outputs a trigger after the satisfaction of a trace condition.

[Address]: Sets address conditions.

[Data]: Sets data conditions.

[Bus/Area]: Sets access type, bus status, and read/write cycle conditions.

[Probe]: Sets the levels (high or low) of the external probe signals (PRB1 to PRB4) as the condition.

[Interrupt]: Sets the levels (high or low) of the IRQ signals as the condition.

[Count]: Sets a satisfaction count condition.

[Delay]: Sets the number of bus cycles delayed after the satisfaction of a trace condition. This function allows you to check the trace information before/after any of the specified conditions are satisfied. When [Don't Care] has been selected, there is no delay.

Notes: 1. The settings to be made on pages [Address], [Data], [Bus/Area], [Probe], [Interrupt], and [Count] are the same as those for on-emulator break conditions. For details on the on-emulator break conditions, refer to section 3.7, Using the Event Points.

2. Set the range in the address range trace so that value of the end address will be larger than that of the start address.

3. Two channels are used in the address range conditional trace. To perform the address range conditional trace, select an odd-numbered channel (2n + 1) for the address range trace and an even-numbered channel (2n + 2) for the conditional trace, respectively, and then check [Combination with the next channel].

3.8.4 Searching for a Trace Record

Use the [Trace Find] dialog box to search for a trace record. To open this dialog box, choose [Find...] from the popup menu.

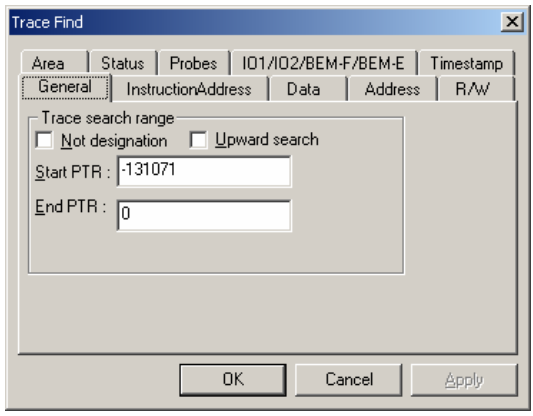


Figure 3.30 [Trace Find] Dialog Box

The [Trace Find] dialog box has the following options:

[General]: Sets the range for searching.

- [Not designation]: Searches for information that does not match the conditions set in other pages when this box is checked.
- [Upward search]: Searches upwards when this box is checked.
- [Start PTR]: Enters a PTR value to start a search.
- [End PTR]: Enters a PTR value to end a search.

[InstructionAddress]: Set an instruction address condition.

- [Don't care]: Detects no address when this box is checked.
- [Value]: Detects the specified instruction address. Enter an address value.

[Data]: Set a data condition.

- [Don't care]: Detects no data when this box is checked.
- [Value]: Detects the specified data. Enter a data value.

[Address]: Set an address condition.

- [Don't care]: Detects no address when this box is checked.
- [Value]: Detects the specified address. Enter an address value.

[R/W]: Select the type of access cycles.

[Don't care]: Detects no read/write condition when this box is checked.

[Setting]: Detects the specified read/write condition.
[RD]: Read cycle
[WR]: Write cycle

[Area]: Select the area being accessed.

[Don't care]: Detects no area condition when this box is checked.

[Setting]: Detects the specified area condition.
[INROM]: ROM
[INRAM]: RAM
[I/O]: I/O
[EXT-32]: 32-bit EXT (external memory)
[EXT-16]: 16-bit EXT (external memory)
[EXT-8]: 8-bit EXT (external memory)
[EMU-32]: 32-bit EMU (emulation memory)
[EMU-16]: 16-bit EMU (emulation memory)
[EMU-8]: 8-bit EMU (emulation memory)

[Status] Select the status of a bus.

[Don't care]: Detects no bus condition when this box is checked.

[Setting]: Detects the specified bus condition.
[PROG]: Prefetch cycle
[DATA]: CPU data access cycle
[DMAC]: DMAC cycle
[DTC]: DTC operation
[STACK]: Stack cycle

[Probes]: Select the status of probe signals.

[Don't care]: Detects no probe signal condition when this box is checked.

[Setting]: Detects the specified probe signal condition.
Don't care: Detects no selected probe condition.
High: The status of the probe signal is high.
Low: The status of the probe signal is low.

[IRQ]: Select the status of IRQ signals.

[Don't care]: Detects no IRQ signal condition when this box is checked.

[Setting]: Detects the specified IRQ signal condition.
Don't care: Detects no selected IRQ condition.
High: The status of the IRQ signal is high.
Low: The status of the IRQ signal is low.

[Timestamp]: Specify the time stamp value for bus cycles.

[Don't care]: Detects no time stamp value when this box is checked.

[Setting]: Detects the specified time stamp value. Enter a time stamp value.
(Every field must be filled in.)

Clicking the [OK] button after setting conditions in those pages stores the settings and starts searching. Clicking the [Cancel] button closes this dialog box without setting of conditions.

When a trace record that matches the search conditions is found, the line for the trace record will be highlighted. When no matching trace record is found, a message dialog box will appear.

Only the trace information that satisfies all the conditions set in above pages will be searched.

If a find operation is successful, selecting [Find Next] from the popup menu will move to the next found item.

3.8.5 Clearing the Trace Information

Select [Clear] from the popup menu to empty the trace buffer that stores the trace information. If several [Trace] windows are open, all [Trace] windows will be cleared as they all access the same buffer.

3.8.6 Saving the Trace Information in a File

Select [Save...] from the popup menu to open the [Save As] file dialog box, which allows the user to save the information displayed in the [Trace] window as a text file. A range can be specified based on the [PTR] number (saving the complete buffer may take several minutes). Note that this file cannot be reloaded into the [Trace] window.

Note: In filtering of trace information, the range to be saved cannot be selected. All the trace information displayed in the [Trace] window after filtering will be saved. Select a filtering range on the [General] page in the [Trace Filter] dialog box if you want to save the selected range. For details on the filtering function, refer to section 3.8.11, Extracting Records from the Acquired Information.

3.8.7 Viewing the [Editor] Window

The [Editor] window corresponding to the selected trace record can be displayed in the following two ways:

- Select a trace record and choose [View Source] from the popup menu.
- Double-click a trace record

The [Editor] or [Disassembly] window opens and the selected line is marked with a cursor.

3.8.8 Trimming the Source

Choose [Trim Source] from the popup menu to remove the white space from the left side of the source.

When the white space is removed, a check mark is shown to the left of the [Trim Source] menu. To restore the white space, choose [Trim Source] while the check mark is shown.

3.8.9 Temporarily Stopping Trace Acquisition

To temporarily stop trace acquisition during execution of the user program, select [Halt] from the popup menu. This stops trace acquisition and updates the trace display. Use this method to check the trace information without stopping execution of the user program.

3.8.10 Restarting Trace Acquisition

To restart trace acquisition being stopped during execution of the user program, select [Restart] from the popup menu.

3.8.11 Extracting Records from the Acquired Information

Use the filtering function to extract the records you need from the acquired trace information. The filtering function allows the trace information acquired by hardware to be filtered by software. Unlike the settings made in the [Trace Acquisition] dialog box for acquiring trace information by conditions, changing the settings for filtering several times to filter the acquired trace information allows easy extraction of necessary information, which is useful for analysis of data. The content of the trace buffer will not be changed even when the filtering function is used. Acquiring useful information as much as possible by the [Trace Acquisition] settings improves the efficiency in analysis of data because the capacity of the trace buffer is limited.

Use the filtering function in the [Trace Filter] dialog box to select a range for filtering.

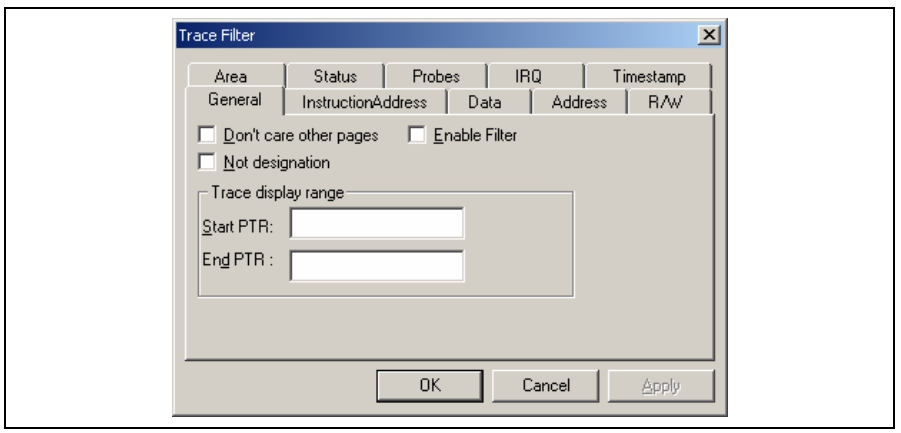


Figure 3.31 [Trace Filter] Dialog Box ([General] Page)

To open the [Trace Filter] dialog box, select [Filter…] from the popup menu.

The [Trace Filter] dialog box has the following pages:

[General]: Sets the range for filtering.

- [Don't care other pages]: Only selects the cycle number when this box is checked. Other options become invalid.
- [Enable Filter]: Enables the filter when this box is checked.
- [No]: Filters information that does not match the conditions set in those pages when this box is checked.
- [Start PTR]: Enter a PTR value to start filtering.
- [End PTR]: Enter a PTR value to end filtering.

[InstructionAddress]: Set an instruction address condition.

- [Don't care]: Detects no address when this box is checked.
- [Value]: Detects the specified instruction address. Enter an address value.
- [Point]: Enter a single value as an instruction address.
- [Range]: Specify an instruction address range.
- [From]: Enter a single address value or the start of the instruction address range.
- [To]: Enter the end address of the instruction address range.

[Data]: Set a data condition.

- [Don't care]: Detects no data when this box is checked.
- [Setting]: Detects the specified data.
- [Point]: Enter a single data value.
- [Range]: Specify a data range.
- [From]: Enter a single data value or the minimum value of the data range.
- [To]: Enter the maximum value of the data range.

[Address]: Set an address condition.

- [Don't care]: Detects no address when this box is checked.
- [Setting]: Detects the specified address.
- [Point]: Enter a single address value.
- [Range]: Specify an address range.
- [From]: Enter a single address value or the start of the address range.
- [To]: Enter a single address or the end of the address range.

[R/W]: Select the type of access cycles.

- [Don't care]: Detects no read/write condition when this box is checked.
- [Setting]: Detects the specified read/write condition.
  [RD]: Detects read cycles.
  [WR]: Detects write cycles.

[Area]: Select the area being accessed.

[Don't care]: Detects no area condition when this box is checked.

[Setting]: Detects the specified area condition.
[INROM]: Detects accesses to the ROM area.
[INRAM]: Detects accesses to the RAM area.
[I/O]: Detects accesses to the I/O area.
[EXT-32]: Detects accesses to the 32-bit EXT (external memory).
[EXT-16]: Detects accesses to the 16-bit EXT (external memory).
[EXT-8]: Detects accesses to the 8-bit EXT (external memory).
[EMU-32]: Detects accesses to the 32-bit EMU (emulation memory).
[EMU-16]: Detects accesses to the 16-bit EMU (emulation memory).
[EMU-8]: Detects accesses to the 8-bit EMU (emulation memory).

[Status] Select the status of a bus.

[Don't care]: Detects no bus condition when this box is checked.

[Setting]: Detects the specified bus condition.
[PROG]: Detects prefetch cycles
[DATA]: Detects CPU data access cycles
[DMAC]: Detects DMAC cycles
[DTC]: Detects DTC operation
[STACK]: Detects stack cycles

[Probes]: Select the status of probe signals.

[Don't care]: Detects no probe signal condition when this box is checked.

[Setting]: Detects the specified probe signal condition.
Don't care: Detects no selected probe condition.
High: The status of the probe signal is high.
Low: The status of the probe signal is low.

[IRQ]: Select the status of IRQ signals.

[Don't care]: Detects no IRQ signal condition when this box is checked.

[Setting]: Detects the specified IRQ signal condition.
Don't care: Detects no selected IRQ condition.
High: The status of the IRQ signal is high.
Low: The status of the IRQ signal is low.

[Timestamp]: Specify the time stamp value for bus cycles.

[Don't care]: Detects no time stamp value when this box is checked.

[Setting]: Detects the specified time stamp value.

[Point]: Specify a single time stamp value.

[Range]: Specify a time stamp range.

[From]: Enter a single time stamp value or the minimum value of the time stamp range.

[To]: Enter the maximum value of the time stamp range.

Set filtering conditions and then press the [OK] button. This starts filtering according to the conditions. Clicking the [Cancel] button closes the [Trace Filter] dialog box, which holds the settings at the time when the dialog box was opened.

In filtering, only the trace information that satisfies one or more filtering conditions set in the above pages will be displayed in the [Trace] window.

Filtering conditions can be changed several times to analyze data because the content of the trace buffer is not changed by filtering.

3.8.12 Calculating the Difference in Time Stamping

Select [Timestamp Difference…] from the popup menu to calculate the time difference between the two points selected by the result of tracing in acquisition of time stamp information.

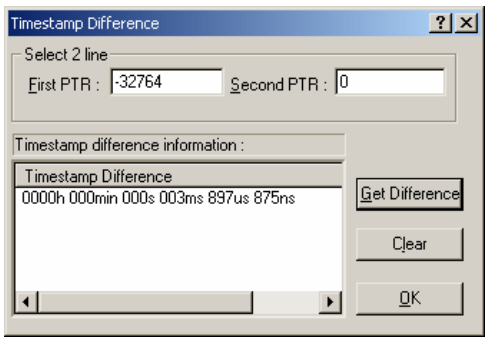


Figure 3.32 [Timestamp Difference] Dialog Box

[Select 2 line]: Select trace records to calculate the time stamp difference.

- [First PTR]: Specifies the first pointer to measure the difference. The pointer of the line selected on the [Trace] window is displayed by default.
- [Second PTR]: Specifies the second pointer to measure the difference.

[Timestamp Difference]: Displays the results of calculation.

[Get Difference]: Calculates the difference between the specified two points and display its result in the [Timestamp Difference] list.

[Clear]: Clears all the results in the [Timestamp Difference] list.

[OK]: Closes the dialog box. All the results in the [Timestamp Difference] list are cleared.

3.8.13 Analyzing Statistical Information

Choose [Statistic] from the popup menu to open the [Statistic] dialog box and analyze statistical information under the specified conditions.

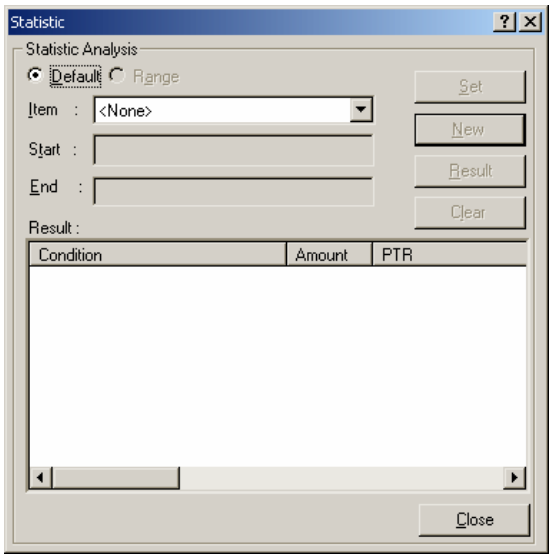


Figure 3.33 [Statistic] Dialog Box

[Statistic Analysis]: Setting required for analysis of statistical information.

[Default]: Sets a single input value or character string.

[Range]: Sets the input value or character string as a range.

[Item]: Sets the item for analysis.

[Start]: Sets the input value or character string. To set a range, the start value must be specified here.

[End]: Specify the end value if a range has been set (only available when [Range] has been selected).

[Set]: Adds a new condition to the current one.

[New]: Creates a new condition.

[Result]: Obtains the result of statistical information analysis.

[Clear]: Clears all conditions and results of statistical information analysis.

[Close]: Closes this dialog box. All the results displayed in the [Result] list will be cleared.

This dialog box allows the user to analyze statistical information concerning the trace information. Set the target of analysis in [Item] and the input value or character string by [Start] and [End]. Click the [Result] button after setting a condition by pressing the [New] or [Add] button to analyze the statistical information and display its result in the [Result] list.

Note: In this emulator, only [PTR] can be set as a range. Each of other items must be specified as a character string. In analysis of statistical information, character strings are compared with those displayed in the [Trace] window. Only those that completely match are counted. Note, however, that this test is not case sensitive. The number of blanks will not be cared either.

3.8.14 Extracting Function Calls from the Acquired Trace Information

To extract function calls from the acquired trace information, select [Function Call…] from the popup menu. The [Function Call Display] dialog box will be displayed.

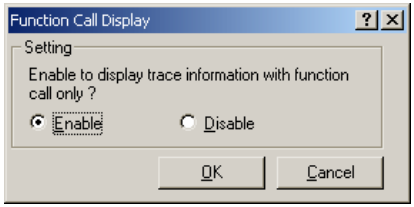


Figure 3.34 [Function Call Display] Dialog Box

[Setting]: Selects whether or not to extract function calls.

[Enable]: Extracts function calls.

[Disable]: Does not extract function calls.

When [Enable] is selected, only the cycles that include function calls are extracted for display from the acquired trace information. The content of the trace buffer is not changed by extraction of function calls. Using this function for the result of the free trace or the trace information that includes function calls allows the user to know the order of function calls.

3.9 Analyzing Performance

Use the performance analysis function to measure the rate of execution time. The performance analysis function does not affect the realtime operation because it measures the rate of execution time in the specified range by using the circuit for measurement of hardware performance included in the emulator.

Select one of the following five modes according to the purpose of measurement.

Table 3.1 Available Measurement Modes

Mode	Description	Purpose
Time Of Specified Range Measurement	Measures the execution time and execution count in the specified range.	Measurement of time taken for processing of functions except for that required for child functions called from the functions.
Start Point To End Point Measurement	Measures the execution time and execution count between the specified addresses.	Measurement of time taken for processing of functions.
Start Range To End Range Measurement	Measures the execution time from a specified range to another specified range.	Measurement of execution time spent from calling of any of sequential subroutines to calling of any of other sequential subroutines in a program that includes subroutines in sequence, such as an assembly program.
Access Count Of Specified Range Measurement	Measures the number of times a specified range is accessed from another specified range.	Measurement of the number of times a global variable is accessed from a specific function.
Called Count Of Specified Range Measurement	Measures the number of times a specified range has called another specified range.	Measurement of the number of times a function is called from a specific function.

Use eight performance channels installed on the circuit for measurement of hardware performance in the emulator for setting of conditions for measurement. Up to eight points can be set.

Note, however, that up to four points can be set in Start Range To End Range Measurement, Access Count Of Specified Range Measurement, or Called Count Of Specified Range Measurement because two sequential points are used for setting a condition in these modes.

Table 3.2 Mode Settings for Measurement

Measurement Mode	Point 1	2	3	4	5	6	7	8
Time Of Specified Range Measurement	O	O	O	O	O	O	O	O
Start Point To End Point Measurement	O	O	O	O	O	O	O	O
Start Range To End Range Measurement	O	—	O	—	O	—	O	—
Access Count Of Specified Range Measurement	O	—	O	—	O	—	O	—
Called Count Of Specified Range Measurement	O	—	O	—	O	—	O	—

Note: O: Available
—: Not available

Note: Only one point is used in Time Of Specified Range Measurement and Start Point To End Point Measurement, while two sequential points are used in Start Range To End Range Measurement, Access Count Of Specified Range Measurement, and Called Count Of Specified Range Measurement. The conditions that have been set will be canceled when switching these modes of different types.

3.9.1 Opening the [Performance Analysis] Window

Choose [View -> Performance -> Performance Analysis] or click the [PA] toolbar button ( ) to open the [Select Performance Analysis Type] dialog box.

Figure 3.35 [Select Performance Analysis Type] Dialog Box

Select [E6000H Performance Analysis] and then click the [OK] button to open the [Performance Analysis] window.

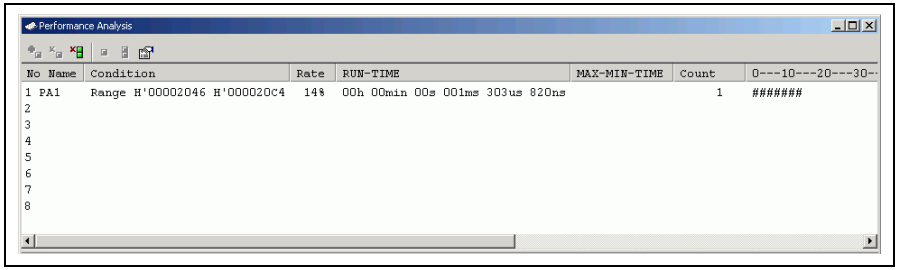


Figure 3.36 [Performance Analysis] Window

This window displays the rate of execution time in the area selected by the user during the last program run in percentages, histogram, or numerical values.

It is possible to hide any column not necessary in the [Performance Analysis] window. Selecting a column you want to hide from the popup menu displayed by clicking the right-hand mouse button on the header column hides that column. To display the hidden column, select the column from the said popup menu again.

3.9.2 Setting Conditions for Measurement

Conditions for measurement can be displayed and changed in the [Performance Analysis] window. Select a point where a condition is to be set, and then select [Set…] from the popup menu to display the [Performance Analysis Properties] dialog box.

Select either from the following five modes as the condition by the [Measurement Method] option:

Table 3.3 Conditions for Measurement (Measurement Method)

[Measurement Method] Option
Time Of Specified Range Measurement
Start Point To End Point Measurement
Start Range To End Range Measurement
Access Count Of Specified Range Measurement
Called Count Of Specified Range Measurement

Set a condition for measurement according to the mode being selected. The parameters to be set depend on the modes.

The [Performance Analysis Properties] window has a support function to enter the address range of a function automatically if the name of the function is entered to set an address range. Entering a function name in the [Input Function Range] dialog box displayed by clicking the […] button on the [Performance Analysis Properties] dialog box automatically enters the address range of the function.

Figure 3.37 [Input Function Range] Window

Notes: 1. Entering the name of an overload function or a class opens the [Select Function] dialog box. Select a function in this dialog box.

2. The addresses figured out are just for reference. In some cases, the end address of a function may be different. Check the last instruction of the function in the [Disassembly] window to correct the value set in [End Address] so that it will be the address of the last instruction (in general, the last instruction of a function is a RTS instruction). A label name or an expression can be entered instead of an address value in boxes where an address should be entered.

(1) Time Of Specified Range Measurement

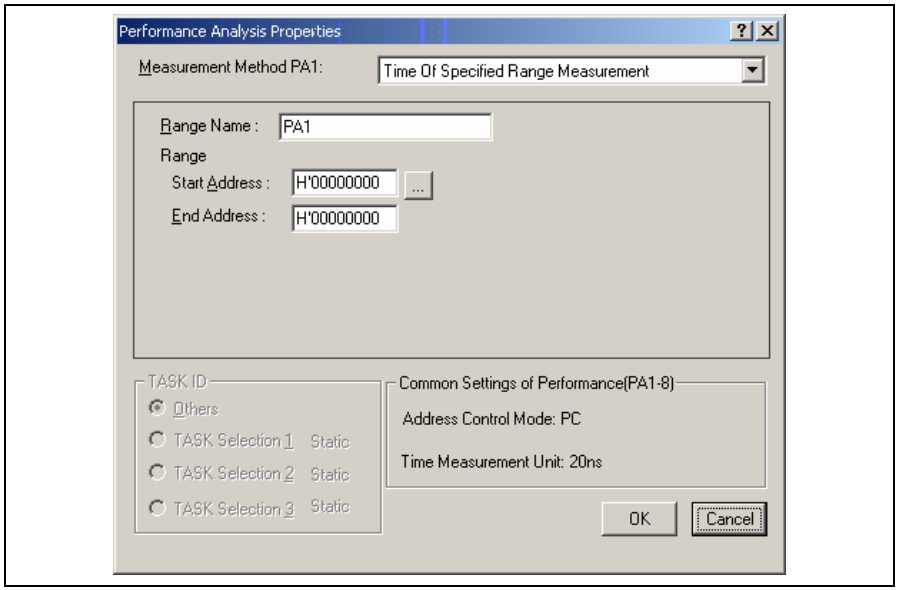


Figure 3.38 Time Of Specified Range Settings

[Range Name]: The name of the range to be measured

[Range]: The range for the Time Of Specified Range Measurement

[Start Address]: Address to start measurement

[End Address]: Address to end measurement

Measures the execution time and the execution count in the range between the start address and end address. Starts measurement with a detected program prefetch in the range specified between the start and end addresses, and then stops with a detected program prefetch out of the specified range. Measurement can be restarted with a detected program prefetch in the specified range. The execution count is incremented every time the program is prefetched at the end address of the specified range. The execution time measured does not include the time spent while being called from the specified range.

(2) Start Point To End Point Measurement

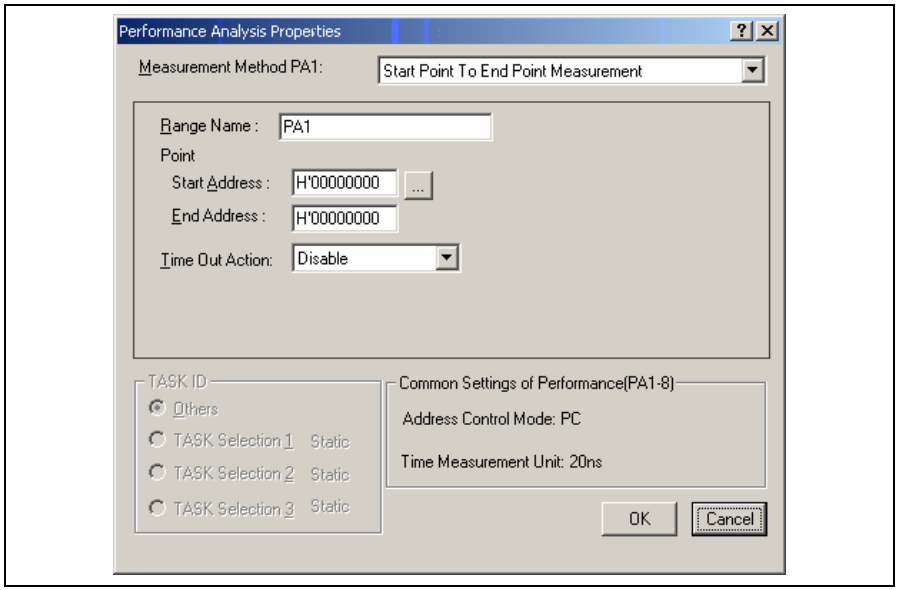


Figure 3.39 Start Point To End Point Measurement Settings

[Range Name]: The name of the range to be measured

[Point]: The range for the Start Point To End Point Measurement

[Start Address]: Address to start measurement

[End Address]: Address to end measurement

[Time Out Action]: The action to take when a timeout or count-out occurs.

Disable: Disables setting of a timeout or count-out value.
Enable: Stops the user program execution when a timeout or count-out occurs.
Trace Stop: Stops trace acquisition when a timeout or count-out occurs.

This is only available for channel 1.

[Time Out]: The timeout value to finish measurement. When the minimum time for measurement is 160 ns, 40 ns, or 20 ns, enter the value as follows.
Example: 1h 2min 3s 123ms 456us 789ns

If the CPU operating mode is target, enter a hexadecimal number in 10 digits.
Example: 123456789A

A break occurs every time a value measured in the specified range exceeds the timeout value (not the total time). This is only available for channel 1.

[Count]: The count-up value used in measurement of the execution count. A break occurs every time the execution count exceeds the count-up value. This is only available for channel 1.

Measures the execution time and the execution count in the range between start address and end address. Starts measurement with a detected program prefetch at the start address, and then stops with a detected program prefetch at the end address. The execution count is incremented every time the program is prefetched at the end address of the specified range. The execution time measured includes the time spent while being called from the specified range. When either from one to four points is selected, the maximum and minimum time in the specified range can be measured.

(3) Start Range To End Range Measurement

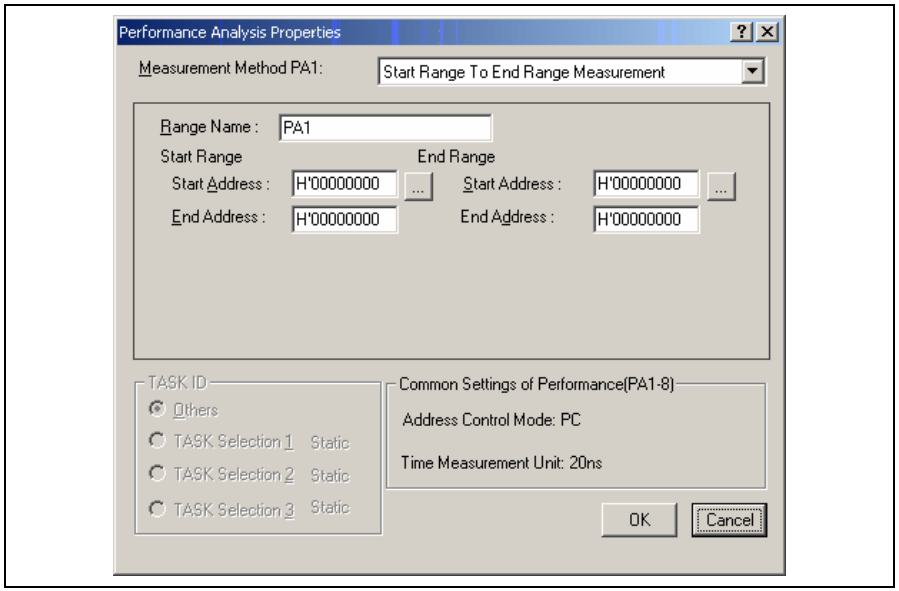


Figure 3.40 Start Range To End Range Measurement Settings

[Range Name]: The name of the range to be measured

[Start Range]: The start range for the Start Range To End Range Measurement

[Start Address]: Start address

[End Address]: End address

[End Range]: The end range for the Start Range To End Range Measurement

[Start Address]: Start address

[End Address]: End address

Starts measurement with a detected prefetch cycle in the specified start address range, and then stops with a detected prefetch cycle in the specified end address range. The execution count is incremented every time the program passes the end address range.

(4) Access Count Of Specified Range Measurement

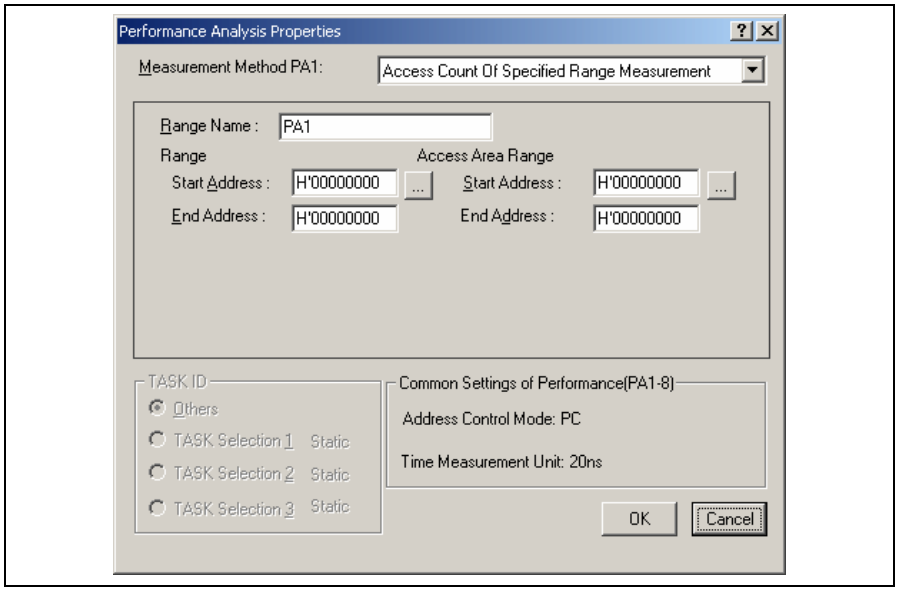


Figure 3.41 Access Count Of Specified Range Measurement Settings

[Range Name]: The name of the range to be measured

[Range]: The range for the Access Count Of Specified Range Measurement

[Start Address]: Start address

[End Address]: End address

[Access Area Range]: The access range for the Access Count Of Specified Range Measurement

[Start Address]: Start address

[End Address]: End address

Measures the number of times the range specified as the access range is accessed from the range specified by the start and end addresses. The execution count in the range is measured with Time Of Specified Range Measurement mode.

(5) Called Count Of Specified Range Measurement

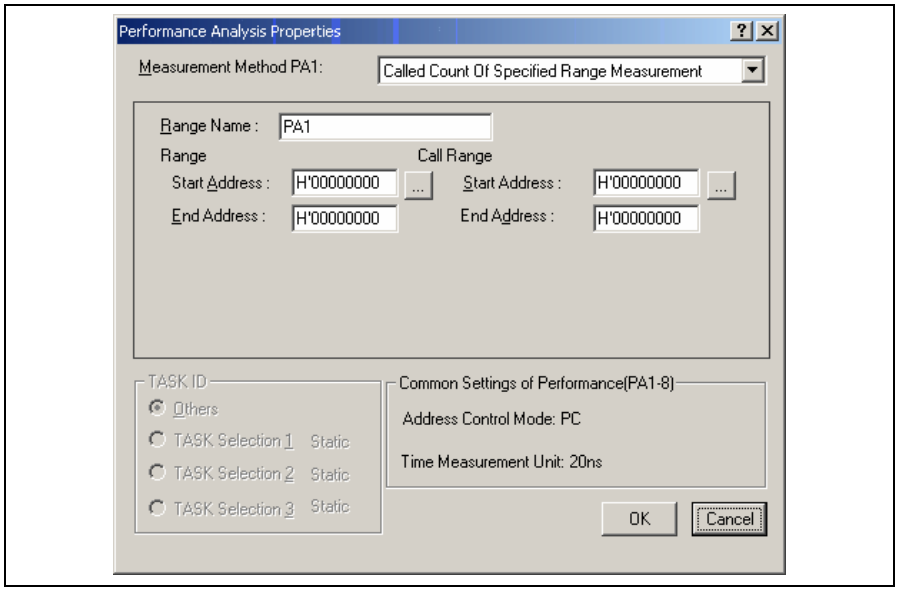


Figure 3.42 Called Count Of Specified Range Measurement Settings

[Range Name]: The name of the range to be measured

[Range]: The range for the Called Count Of Specified Range Measurement

[Start Address]: Start address

[End Address]: End address

[Call Range]: The range for the Called Count Of Specified Range Measurement. As the call range, specify the start and end addresses of the selected subroutine.

[Start Address]: Start address

[End Address]: End address

Measures the number of times the range specified as the call range is called from the range specified by the start and end addresses. The execution time in the specified range can be measured with Time Of Specified Range Measurement mode. As the call range, specify the start and end addresses of the selected subroutine.

3.9.3 Starting Performance Data Acquisition

Executing the user program clears the result of previous measurement and automatically starts measuring the rate of execution time according to the conditions that have been set. Stopping the user program displays the result of measurement in the [Performance Analysis] window.

3.9.4 Deleting a Measurement Condition

Select [Reset] from the popup menu with a measurement condition selected to delete the condition.

3.9.5 Deleting All Measurement Conditions

Choose [Reset All] from the popup menu to delete all the conditions that have been set.

3.10 Profiling Function

3.10.1 Enabling the Profile

Choose [View->Performance->Profile] to open the [Profile] window. Choose [Enable Profiler] from the popup menu of the [Profile] window. The item on the menu will be checked.

3.10.2 Specifying Measuring Mode

You can specify whether to trace functions calls while profile data is acquired. When function calls are traced, the relations of function calls during user program execution are displayed as a tree diagram. When not traced, the relations of function calls cannot be displayed, but the time for acquiring profile data can be reduced.

To stop tracing function calls, choose [Disable Tree (Not traces function call)] from the popup menu in the [Profile] window (a check mark is shown to the left of the menu item).

When acquiring profile data of the program in which functions are called in a special way, such as task switching in the OS, stop tracing function calls.

3.10.3 Executing the Program and Checking the Results

After the user program has been executed and execution has been halted, the results of measurement are displayed in the [Profile] window.

The [Profile] window has two sheets; a [List] sheet and a [Tree] sheet.

3.10.4 [List] Sheet

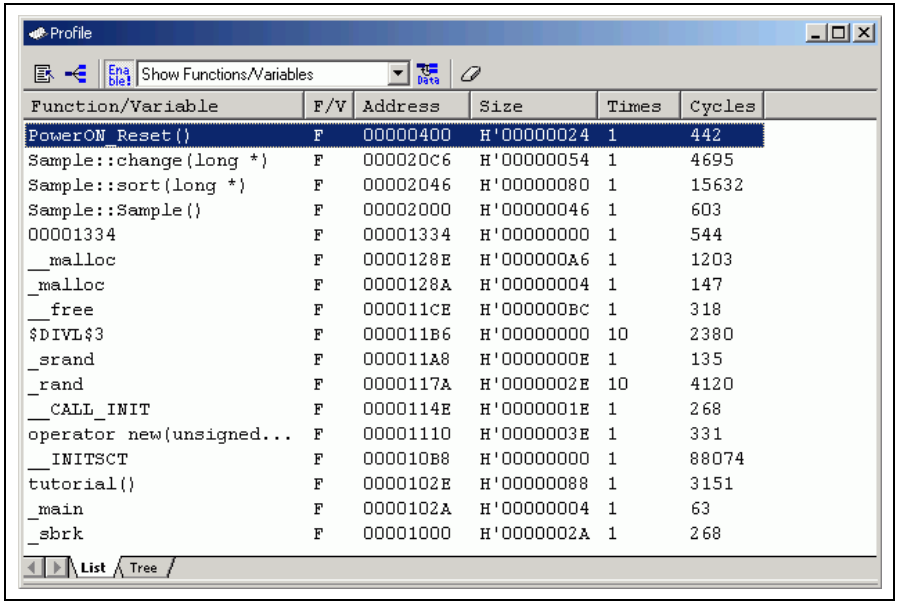


Figure 3.43 [Profile] Window ([List] Sheet)

This window displays the address and size of a function or a global variable, the number of times the function is called or the global variable is accessed, and profile data.

When the column header is clicked, data are sorted in alphabetic or numeric ascending/descending order.

Double-clicking the [Function/Variable] or [Address] column displays the source program of the address in the line. Right-clicking on the mouse within the window displays a popup menu. For details on this popup menu, refer to section 3.10.5, [Tree] Sheet.

Note: For notes on the profiling function, refer to section 5.9, Profiling Function.

3.10.5 [Tree] Sheet

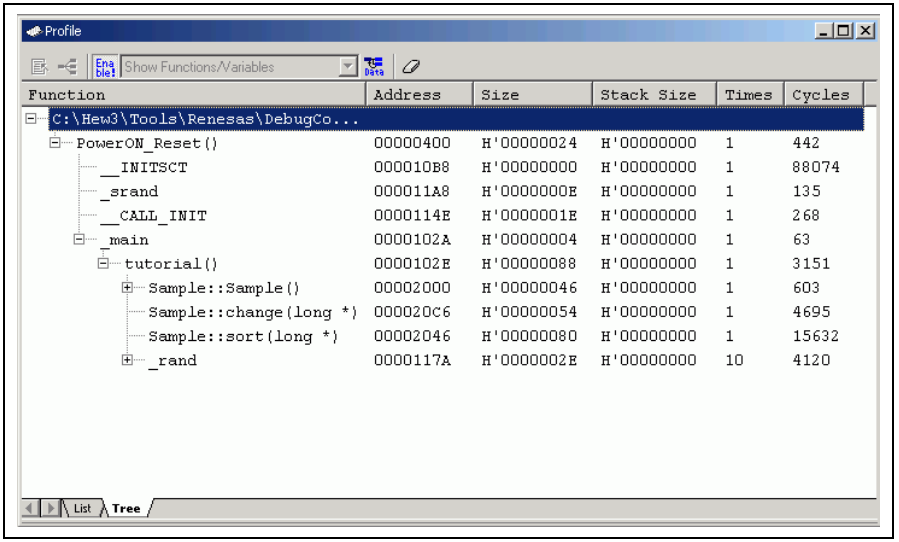


Figure 3.44 [Profile] Window ([Tree] Sheet)

This window displays the relation of function calls in a tree structure. Displayed contents are the address, size, stack size, and number of function calls and execution cycles. The stack size and number of function calls are values when the function is called.

The [Tree] sheet is only available when [Not trace the function call] is not checked in the popup menu of the [Profile] window.

Double-clicking a function in the [Function] column expands or reduces the tree structure display. The expansion or reduction is also provided by the “+” or “-” key. Double-clicking the [Address] column displays the source program of the specific address.

Right-clicking on the mouse within the window displays a popup menu. Supported menu options are described in the following sections:

- **View Source**

Displays the source program or disassembled memory contents for the address in the selected line.

- **View Profile-Chart**

Displays the [Profile-Chart] window focused on the function in the specified line.

- **Enable Profiler**

Toggles acquisition profile data. When profile data acquisition is active, a check mark is shown to the left of the menu text.

- **Not trace the function call**

Stops tracing function calls while profile data is acquired. This menu is used when acquiring profile data of the program in which functions are called in a special way, such as task switching in the OS.

To display the relation of function calls in the [Tree] sheet of the [Profile] window, acquire profile data without selecting this menu. In addition, do not select this menu when optimizing the program by the optimizing linkage editor using the acquired profile information file.

- **Find…**

Displays the [Find Text] dialog box to find a character string in the [Function] column. Search is started by inputting a character string to be found in the edit box and clicking [Find Next] or pressing the Enter key.

- **Clear Data**

Clears the number of times functions are called and profile data. Data in the [Profile] window's [List] sheet and the [Profile-Chart] window are also cleared.

- **Output Profile Information Files…**

Displays the [Save Profile Information Files] dialog box. Profiling results are saved in a profile information file (.pro extension). The optimizing linkage editor optimizes user programs according to the profile information in this file. For details of the optimization using the profile information, refer to the manual of the optimizing linkage editor.

Note: If profile information has been acquired by selecting the [Not trace the function call] menu, the program cannot be optimized by the optimizing linkage editor.

- **Output Text File…**

Displays the [Save Text of Profile Data] dialog box. Displayed contents are saved in a text file.

- **Setting**

This menu has the following submenus (the menus available only in the [List] sheet are also included).

1. Show Functions/Variables

Displays both functions and global variables in the [Function/Variable] column.

2. Show Functions

Displays only functions in the [Function/Variable] column.

3. Show Variables

Displays only global variables in the [Function/Variable] column.

4. Only Executed Functions

Only displays the executed functions. If a stack information file (.sni extension) output from the optimizing linkage editor does not exist in the directory where the load module is located, only the executed functions are displayed even if this check box is not checked.

5. Include Data of Child Functions

Sets whether or not to display information for a child function called in the function as profile data.

- **Properties...**

This popup menu is unavailable in the H8SX E6000H emulator.

3.11 [Profile-Chart] Window

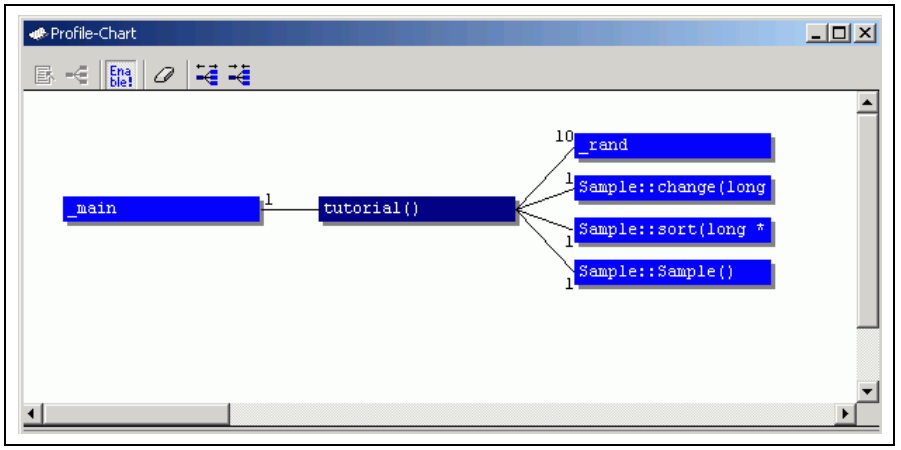


Figure 3.45 [Profile-Chart] Window

This window displays the relation of calls for a specific function. This window displays the calling relation for the function specified in the [List] sheet or [Tree] sheet in the [Profile] window. The specified function is displayed in the middle, the calling function on the left side, and the called function on the right side. Values beside the calling and called functions show the number of times the function has been called.

Right-clicking on the mouse within the window displays a popup menu. Supported menu options are described in the following sections.

- **View Source**

Displays the source program or disassembled memory contents for the address of the function on which the cursor is placed when the right-hand mouse button is clicked. If the cursor is not placed on a function when the right-hand mouse button is clicked, this menu option is displayed in gray characters.

- **View Profile-Chart**

Displays the [Profile-Chart] window for the specific function on which the cursor is placed when the right-hand mouse button is clicked. If the cursor is not placed on a function when the right-hand mouse button is clicked, this menu option is displayed in gray characters.

- **Enable Profiler**

Toggles acquisition of profile data. When profile data acquisition is active, a check mark is shown to the left of the menu text.

- **Clear Data**

Clears the number of times functions are called and profile data. Data in the [List] sheet and [Tree] sheet in the [Profile] window are also cleared.

- **Multiple View**

If the [Profile-Chart] window is going to be opened when it has already been opened, selects whether another window is to be opened or the same window is to be used to display data. When a check mark is shown to the left side of the menu text, another window is opened.

- **Output Profile Information File…**

Displays the [Save Profile Information File] dialog box. Profiling results are saved in a profile information file (.pro extension). The optimizing linkage editor optimizes user programs according to the profile information in this file. For details of the optimization using the profile information, refer to the manual of the optimizing linkage editor.

- **Expands Size**

Expands spaces between each function. The “+” key can also be used to expand spaces.

- **Reduces Size**

Reduces spaces between each function. The “-” key can also be used to reduce spaces.

3.12 RTOS Extension Function

The RTOS extension function supports the debugging of RTOS tasks. The function is implemented by setting the IDs of RTOS tasks at particular addresses.

This extends the following emulator functions.

- Step operation
- Tracing
- Performance analysis
- On-emulator break

3.12.1 [RTOS Support Function Configuration Properties] Dialog Box

Select [Option -> Emulator -> RTOS Support Function] or click the [RTOS Support Function] toolbar button (). This opens the [RTOS Support Function Configuration Properties] dialog box. Setting this dialog box for a task makes the extended functions applicable to that task.

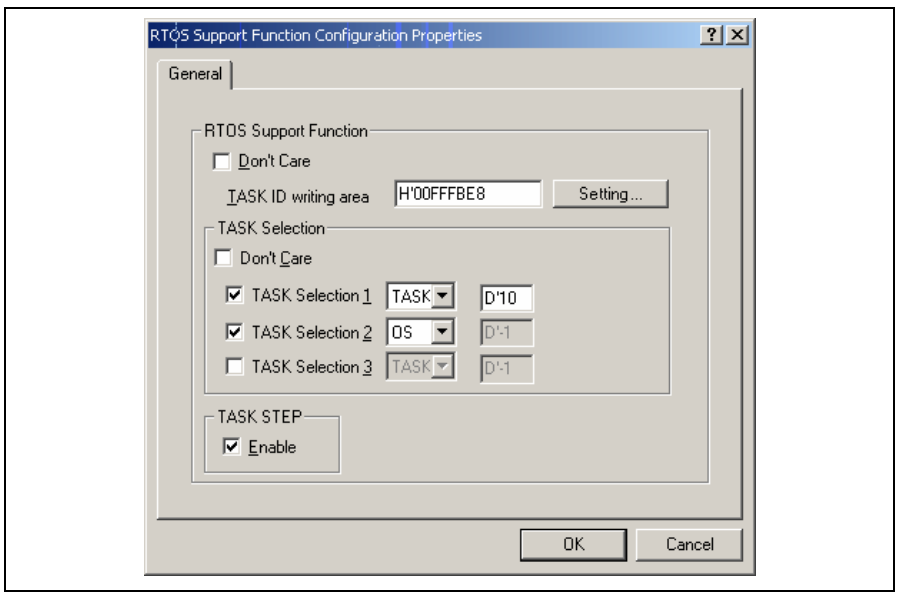


Figure 3.46 [RTOS Support Function Configuration Properties] Dialog Box

[RTOS Support Function]: Sets up the RTOS extension function.

[Don't Care]: Disables the RTOS extension function.

[TASK ID writing area]: Specifies the address where the task ID is written to in the RTOS.

[TASK ID writing area] input edit box: Specifies the address where the task IDs are written. Set an address within the on-chip I/O area that does not contain I/O registers.

[Setting...]: When using the kernel library for E6000H RTOS-debugging support on the Renesas' HI1000/4, searches for the address to which the area for E6000H debugging has been allocated.

[TASK Selection]: Selects task IDs for use in setting conditions for the on-emulator break, tracing, performance analysis, and the display of trace information.

[Don't Care]: Disables [TASK Selection].

[TASK Selection N] (N = 1 to 3): Enables the TASK Selection N condition for use as an on-emulator breakpoint. Setting a TASK Selection N condition as an on-emulator breakpoint, etc. specifies the task ID or OS, as specified in [TASK Selection N], as a condition.

Drop-down list for selecting a task ID or OS: Select the task ID or OS. A task ID of 0 selects the OS.

Edit box for entering a task ID: Set a task ID within the range from D'1 to D'255.

[TASK STEP]: Specifies step operation in task units.

[Enable]: Selection enables step operation in task units.

3.12.2 Task Step Functions

When [Enable] under [TASK STEP] in the [RTOS Support Function Configuration Properties] dialog box, (opened by selecting [Option -> Emulator -> RTOS Support Function]) is selected, the following step operation functions become applicable to the selected tasks.

- Step in : Steps into the selected tasks.
- Step over : Steps over the selected tasks.
- Step out : Steps out of the selected tasks.

3.12.3 Functions Made Available by [TASK Selection]

(1) Display of the [TASK Selection] Settings

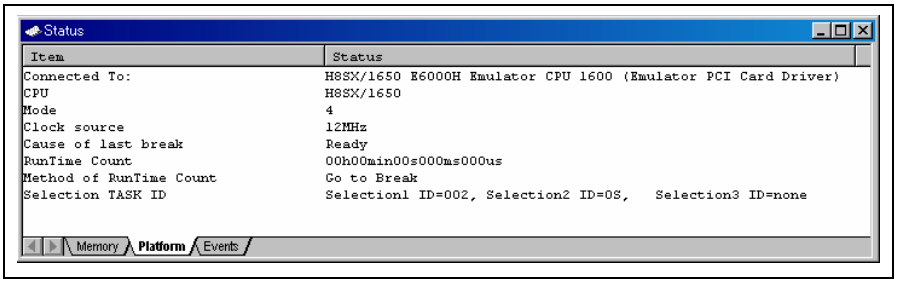


Figure 3.47 [Status] Window

[Platform] sheet: Includes information on the emulation environment, such as the CPU type and operating mode.

[Selection TASK ID]: Displays the values set for [TASK Selection 1] to [TASK Selection 3] in the [RTOS Support Function Configuration Properties] dialog box.

(2) Display of Trace Information ([Trace] Window)

The [TASK] is added to the items for display.

[TASK]: When a task selected in [TASK Selection 1] to [TASK Selection 3] of the [RTOS Support Function Configuration Properties] dialog box is being executed, its ID is displayed here. When the task ID is 0, [OS] is displayed here. During the execution of task IDs other than those specified for [TASK Selection 1] to [TASK Selection 3], [OTHER] is displayed here.

(3) Trace Information Acquisition Condition ([Trace Acquisition Condition Channel n] Dialog Box)

When [TASK Selection] has been set up, the condition settings change in the following ways.

[Address] page in the [Trace Acquisition Condition Channel n] dialog box:

[Address]: Sets the address-bus condition.

[Outside Range]: [Outside Range] is disabled.

[TASK ID]: [TASK ID] is added to the [Address] page. The [TASK Selection 1] to [TASK Selection 3] settings in the [RTOS Support Function Configuration Properties] dialog box are selectable as conditions. To set values other than those specified in [RTOS Support Function Configuration Properties] as a condition, select [OTHER] here.

3.12.4 Performance Measurement (Conditions for Measurement)

The following conditions are included among those selectable under the [Measurement Method] option for measurement of execution.

Setting the [Measurement Method] option:

[Performance Analysis Properties] page: Sets a condition for measurement.

[TASK ID]:	Allows specification of the values set for [TASK Selection 1] to [TASK Selection 3] in the [RTOS Support Function Configuration Properties] dialog box as conditions. To set values other than those specified in [RTOS Support Function Configuration Properties] as a condition, select [OTHER] here.

3.12.5 Event Point (On-Emulator Breakpoint)

When selecting [TASK Selection], the condition is changed as follows.

[Address] page in the [On Emulator Break Channel n] dialog box:

[Address]:	Sets the address bus condition.
[Outside Range]:	[Outside Range] is disabled.
[TASK ID]:	[TASK ID] is added to the [Address] page. The [TASK Selection 1] to [TASK Selection 3] settings in the [RTOS Support Function Configuration Properties] dialog box are selectable as conditions. To set values other than those specified in [RTOS Support Function Configuration Properties] as a condition, select [OTHER] here.

Note: For values set for the RTOS extension function, refer to section 5.11, RTOS Extension Function.

Section 4 Tutorial

4.1 Introduction

This section describes the main functions of the emulator by using a tutorial program.

The tutorial program is based on the C++ program that sorts ten random data items in ascending or descending order. The tutorial program performs the following actions:

- The `main` function repeatedly calls the `tutorial` function to repeat sorting.
- The `tutorial` function generates random data to be sorted and calls the `sort` and `change` functions in that order.
- The `sort` function enters the array where the random data generated by the `tutorial` function are stored, and sorts them in the ascending order.
- The `change` function then sorts the array, which was sorted in ascending order by the `sort` function, in descending order.

The file `tutorial.cpp` contains the source code for the tutorial program. The file `Tutorial.abs` is a compiled load module in the Dwarf2 format.

Notes: 1. After recompilation, the addresses may differ from those given in this section.
2. This section describes general usage examples for the emulator. For the particular specifications of each product, refer to section 3, Debugging, or the online help.
3. The operation address of `Tutorial.abs` attached to each product differs depending on the product. Replace the address used in this section with the correct address in each product after checking that it is placed on the corresponding line of the source program.
4. In this tutorial, the H8SX E6000H emulator is taken as an example. File paths or the appearance of figures differ depending on the product.

4.2 Running the High-performance Embedded Workshop

Open a workspace by following the procedure listed in section 2.1.3, Selecting an Existing Workspace.

Select the following directory.
OS installation drive \Workspace\Tutorial\E6000H\1650

Note: The directory mentioned above cannot be specified depending on the version of the software. In such cases, specify the following directory instead.

High-performance Embedded Workshop installation destination directory
\Tools\Renesas\DebugComp\Platform\E6000H\1650\Tutorial

Then select the file indicated below.

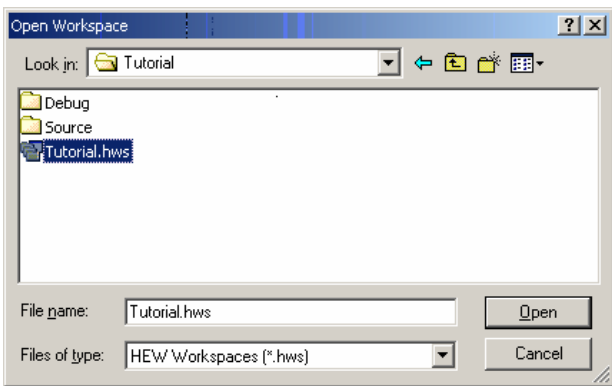


Figure 4.1 [Open Workspace] Dialog Box

4.3 Connecting the Emulator

4.3.1 Selecting a Session

The selectable session depends on the emulator. Select the session suitable for the emulator in use.

Table 4.1 Selectable Sessions

Emulator	Session
HS1527KEPH60H	sessionh8sx_1527_e6000h_emulator
HS1527REPH60H	sessionh8sx_1527r_e6000h_emulator
HS1544EPH60H	sessionh8sx_1544_e6000h_emulator
HS1650EPH60H	sessionh8sx_1650_e6000h_emulator
HS1650EPH60H + HS1653ECN61H	sessionh8sx_1653_e6000h_emulator
HS1650EPH60H + HS1664ECH61H	sessionh8sx_1663_e6000h_emulator

Figure 4.2 Selecting a Session

4.3.2 Connecting the Emulator

Click the [Connect] toolbar button from [Debug] to connect the emulator.

Figure 4.3 Connecting the Emulator

4.4 Downloading the Tutorial Program

4.4.1 Downloading the Tutorial Program

Download the object program to be debugged.

- Select [Download module] from [Tutorial.abs] of [Download modules].

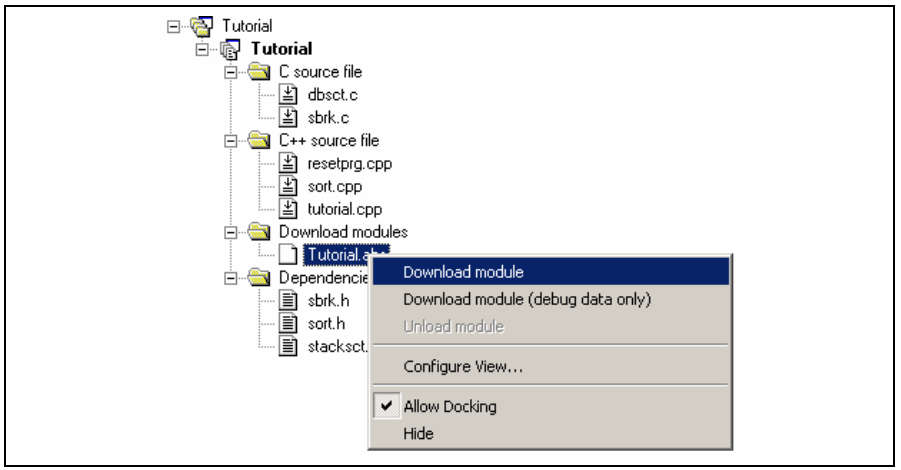


Figure 4.4 Downloading the Tutorial Program

4.4.2 Displaying the Source Program

The High-performance Embedded Workshop allows the user to debug a user program at the source level.

- Double-click [Tutorial.cpp] under [C++ source file].

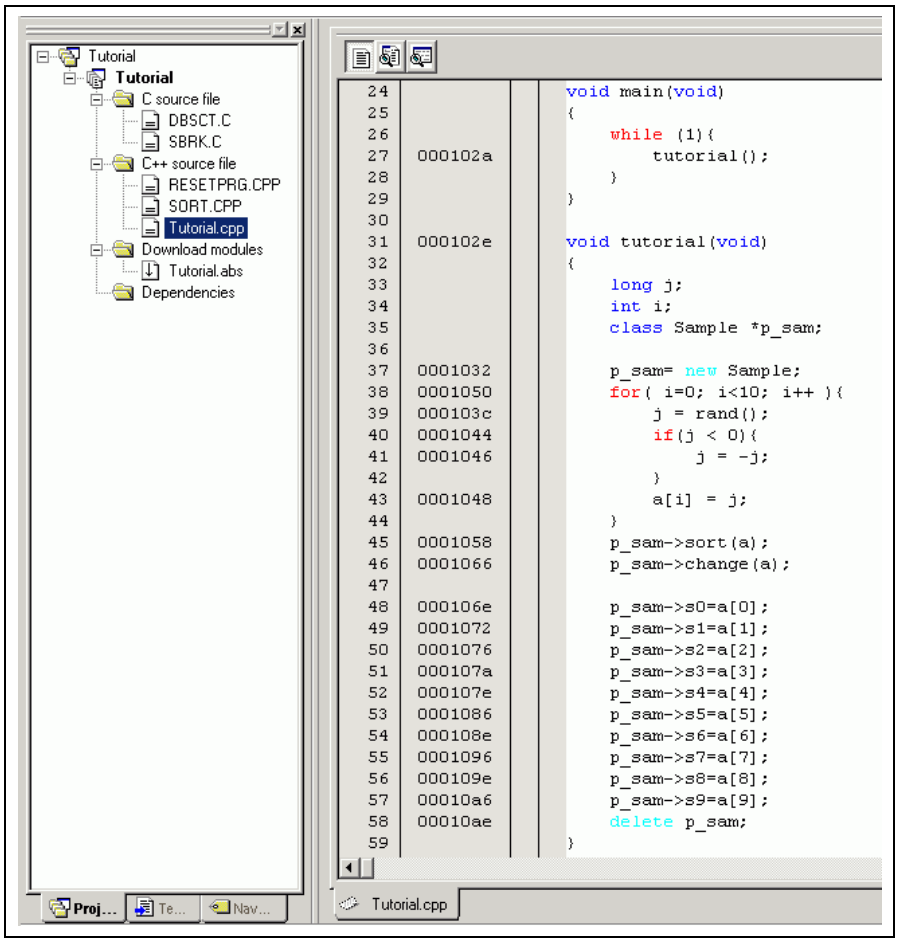


Figure 4.5 [Editor] Window (Displaying the Source Program)

- Select a font and size that are legible if necessary.

Initially the [Editor] window shows the start of the user program, but the user can use the scroll bar to scroll through the user program and look at the other statements.

4.5 Setting a Software Breakpoint

A software breakpoint is a simple debugging function.

The [Editor] window provides a very simple way of setting a software breakpoint at any point in a program. For example, to set a software breakpoint where the `sort` function is called:

- Select by double-clicking the [S/W Breakpoints] column on the line containing the `sort` function call.

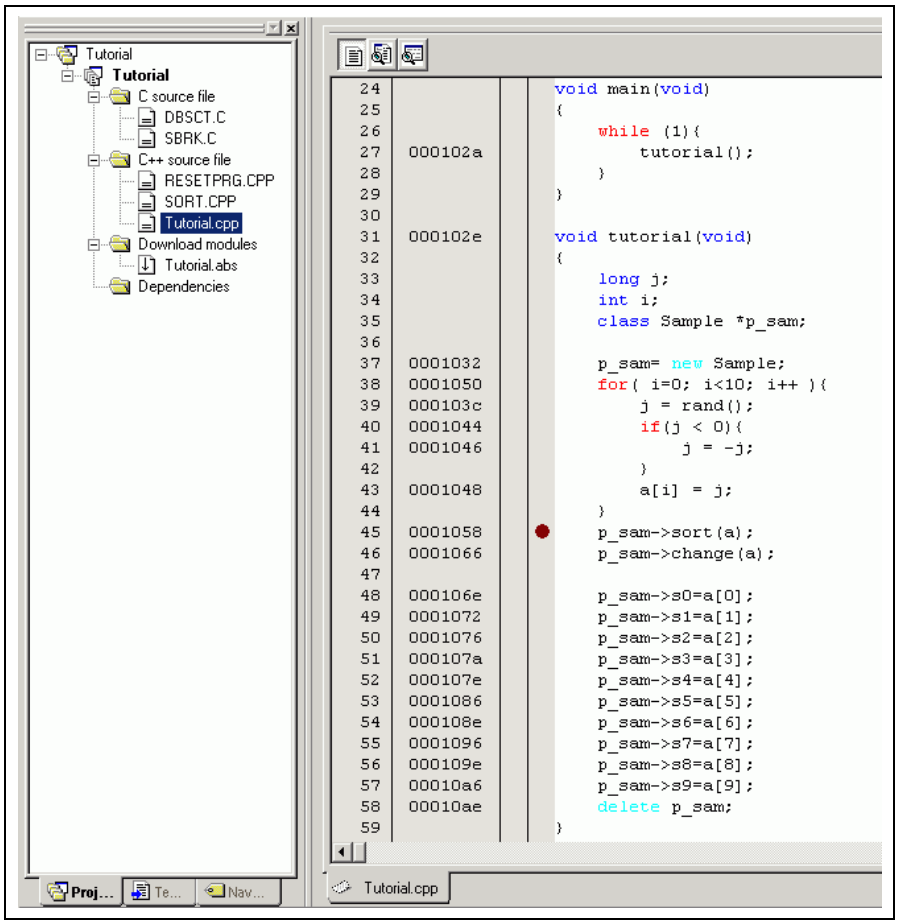


Figure 4.6 [Editor] Window (Setting a Software Breakpoint)

The symbol • will appear on the line containing the `sort` function. This shows that a software breakpoint has been set.

4.6 Setting Registers

Set a value in the program counter before executing the program.

- Select [Registers] from the [CPU] submenu of the [View] menu or click the [Register] toolbar button ( ) to display the [Register] window.

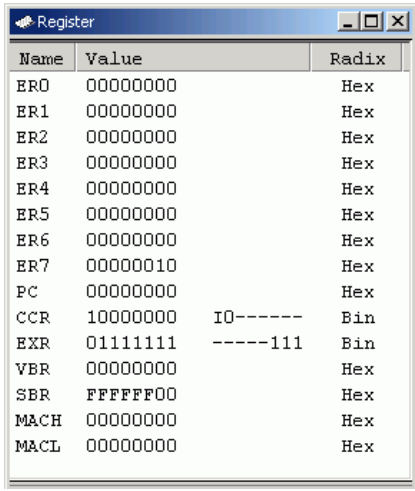


Figure 4.7 [Register] Window

- To change the value of the program counter (PC), double-click on the PC value area in the [Register] window with the mouse. The following dialog box is then displayed, and the value can be changed. Set the program counter to H'00000400 in this tutorial program, and click the [OK] button.

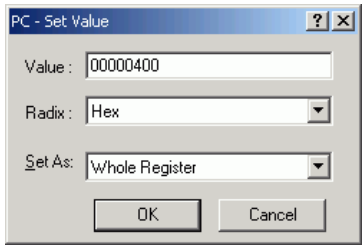


Figure 4.8 [Register] Dialog Box (PC)

4.7 Executing the Program

Execute the program as described in the following:

- To execute the program, select [Go] from the [Debug] menu, or click the [Go] button on the toolbar.

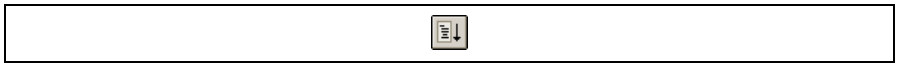


Figure 4.9 [Go] Button

While the program is executed, the current address bus value and the operating state of the MCU are displayed on the status bar.

The program will be executed up to the software breakpoint that has been set, and an arrow will appear on the [S/W Breakpoints] column in the [Editor] window to show the position where the program has halted, with the message `[Break = Software Break]` in the status bar.

Note: When the source file is displayed after a break, a path of the source file may be inquired. The location of the source file is as follows:

OS installation drive \Workspace\Tutorial\E6000H\1650\Source

The directory mentioned above cannot be specified depending on the version of the software. In such cases, specify the following directory instead.

High-performance Embedded Workshop installation destination directory
\Tools\Renesas\DebugComp\Platform\E6000H\1650\Source

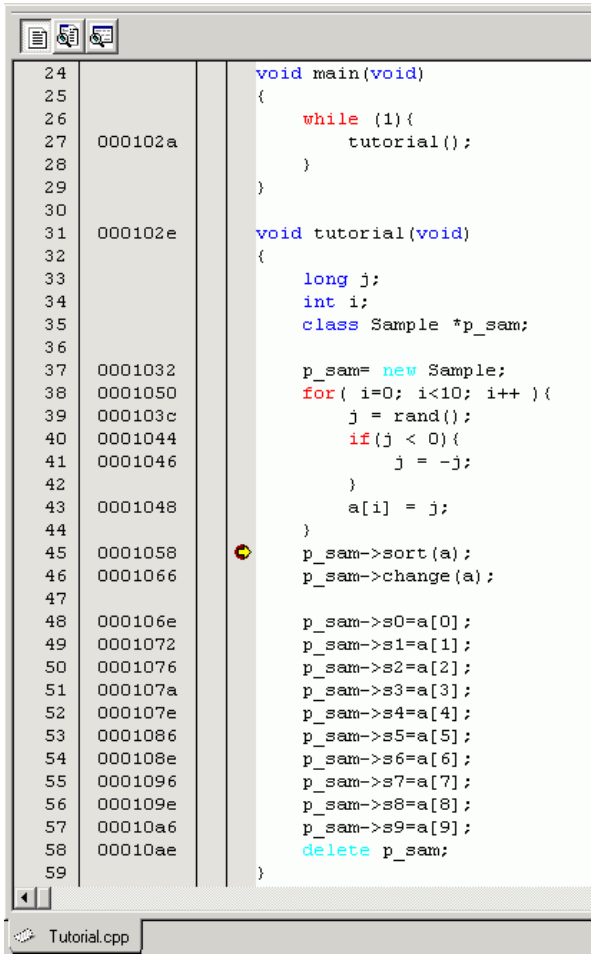


Figure 4.10 [Editor] Window (Break Status)

The user can see the cause of the break that occurred last time in the [Status] window.

- Select [Status] from the [CPU] submenu of the [View] menu or click the [Status] toolbar button ( ). After the [Status] window is displayed, open the [Platform] sheet, and check the `Status` of `Cause of last break`.

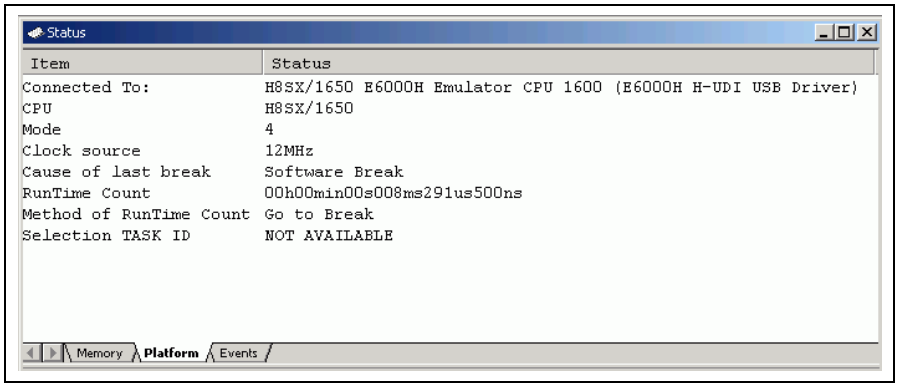


Figure 4.11 [Status] Window

Note: The items that can be displayed in this window differ depending on the product. For the items that can be displayed, refer to section 3, Debugging, or the online help.

4.8 Reviewing Breakpoints

The user can see all the breakpoints set in the program in the [Event] window.

- Select [Eventpoints] from the [Code] submenu of the [View] menu or click the [Eventpoints] toolbar button (). The [Event] window is displayed. Select the [Software] sheet.

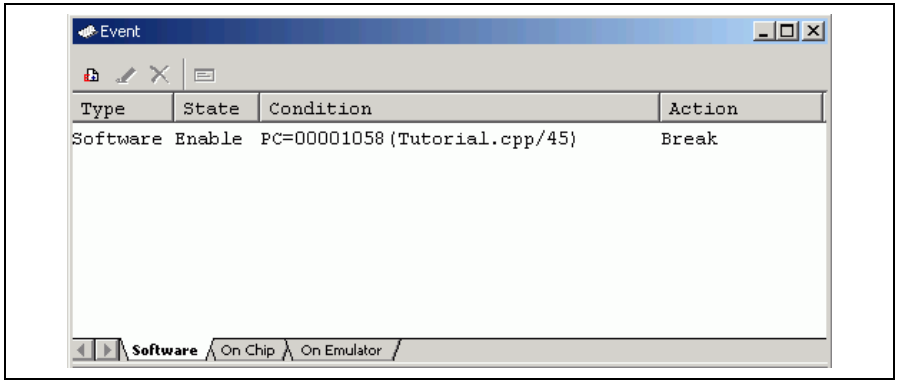


Figure 4.12 [Event] Window

The popup menu, opened by clicking the [Event] window with the right-hand mouse button, allows the user to set or change breakpoints, define new breakpoints, and delete, enable, or disable breakpoints.

4.9 Referring to Symbols

The [Label] window can be used to display the information on symbols in modules.

Select [Label] from the [Symbol] submenu of the [View] menu. The [Label] window is displayed so that the user can refer to the addresses of symbols in modules.

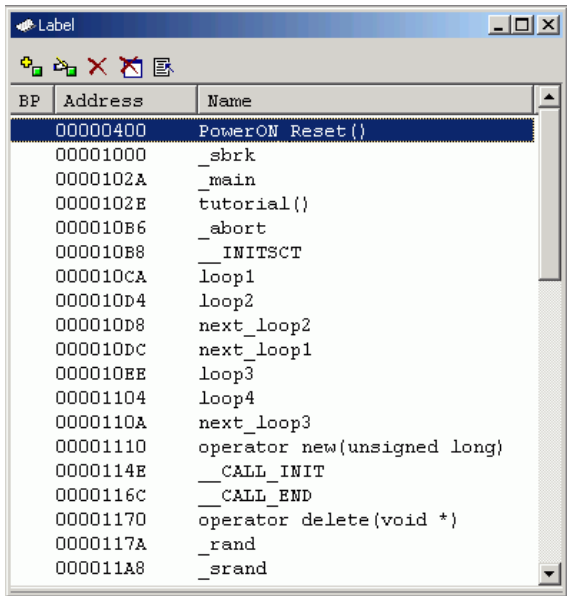


Figure 4.13 [Label] Window

4.10 Viewing Memory

When the label name is specified, the user can view the memory contents that the label has been registered in the [Memory] window. For example, to view the memory contents corresponding to `_main` in byte size:

- Select [Memory …] from the [CPU] submenu of the [View] menu or click the [View Memory] toolbar button ( ) to open the [Format] dialog box. Enter ***_main*** in the [Begin] edit box and ***+ff*** in the [End] edit box, respectively, and select `Byte` in the [Format] combo box.

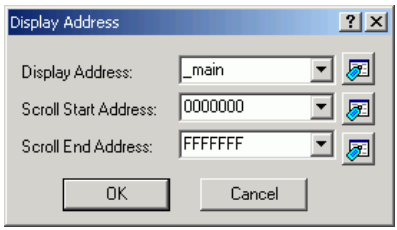


Figure 4.14 [Format] Dialog Box

- Click the [OK] button. The [Memory] window showing the selected area of memory is displayed.

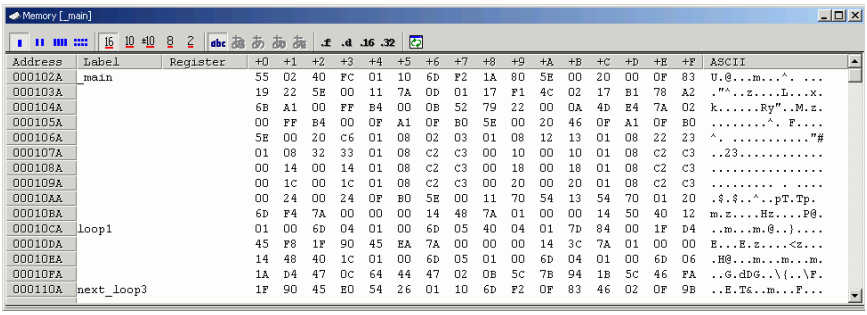


Figure 4.15 [Memory] Window

4.11 Watching Variables

As the user steps through a program, it is possible to watch that the values of variables used in the user program are changed. For example, set a watch on the long-type array `a` declared at the beginning of the program, by the following procedure:

- Click the left of displayed array `a` in the [Editor] window to place the cursor.
- Select [Instant Watch...] with the right-hand mouse button.

The following dialog box will be displayed.

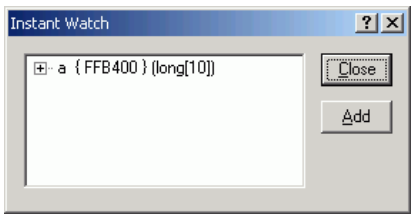


Figure 4.16 [Instant Watch] Dialog Box

- Click the [Add] button to add a variable to the [Watch] window.

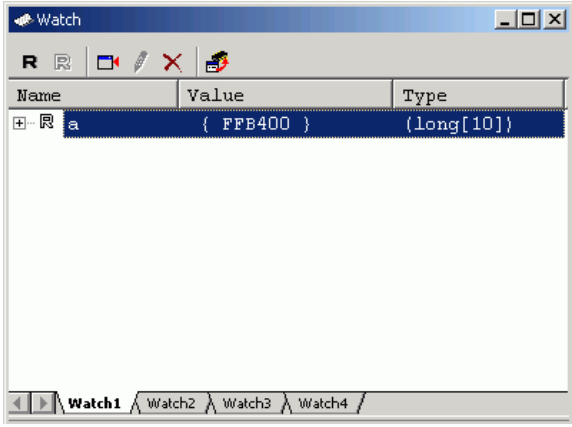


Figure 4.17 [Watch] Window (Displaying the Array)

The user can also add a variable to the [Watch] window by specifying its name.

- Click the [Watch] window with the right-hand mouse button and select [Add Watch…] from the popup menu.

The following dialog box will be displayed.

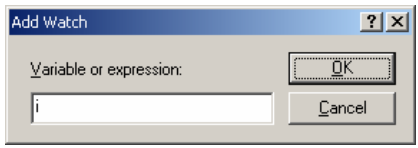


Figure 4.18 [Add Watch] Dialog Box

- Enter variable **i** to [Variable or expression] edit box and click the [OK] button.

The [Watch] window will now also show the int-type variable i.

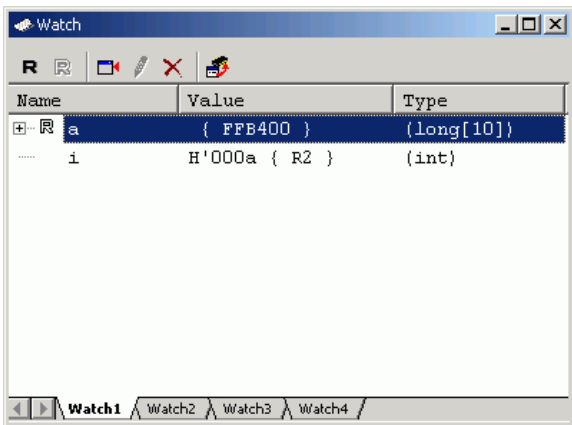


Figure 4.19 [Watch] Window (Displaying the Variable)

The user can click mark '+' at the left side of array a in the [Watch] window to watch all the elements.

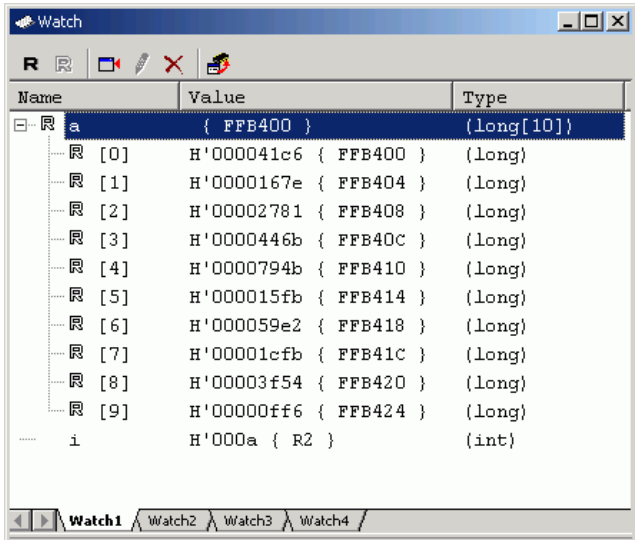

Watch
Name
Value
Type
a { FFB400 } (long[10])
[0] H'000041c6 { FFB400 } (long)
[1] H'0000167e { FFB404 } (long)
[2] H'00002781 { FFB408 } (long)
[3] H'0000446b { FFB40C } (long)
[4] H'0000794b { FFB410 } (long)
[5] H'000015fb { FFB414 } (long)
[6] H'000059e2 { FFB418 } (long)
[7] H'00001cfb { FFB41C } (long)
[8] H'00003f54 { FFB420 } (long)
[9] H'00000ff6 { FFB424 } (long)
i H'000a { R2 } (int)
Watch1
Watch2
Watch3
Watch4

Figure 4.20 [Watch] Window (Displaying Array Elements)

4.12 Displaying Local Variables

The user can display local variables in a function by using the [Locals] window. For example, we will examine the local variables in the `tutorial` function, which declares local variables `j`, `i`, and `p_sam`.

- Select [Locals] from the [Symbol] submenu of the [View] menu. The [Locals] window is displayed.

The [Locals] window shows the local variables in the function currently pointed to by the program counter, along with their values. Note, however, that the [Locals] window is initially empty because local variables are yet to be declared.

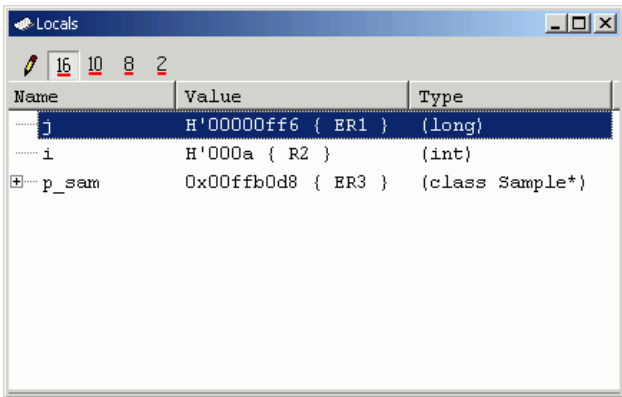


Figure 4.21 [Locals] Window

The user can click mark '+' at the left side of class instance `p_sam` in the [Locals] window to watch all the elements. View the elements of class instance `p_sam` before and after the execution of the `sort` function and check that the random data is sorted in the descending order.

4.13 Stepping Through a Program

The High-performance Embedded Workshop provides a range of step menu commands that allow efficient program debugging.

Table 4.2 Step Options

Menu Command	Description
Step In	Executes each statement, including statements within functions.
Step Over	Executes a function call in a single step.
Step Out	Steps out of a function, and stops at the statement following the statement in the program that called the function.
Step…	Steps the specified times repeatedly at a specified rate.

4.13.1 Executing the [Step In] Command

The [Step In] command steps into the called function and stops at the first statement of the called function.

- To step through the `sort` function, select [Step In] from the [Debug] menu, or click the [Step In] button on the toolbar.

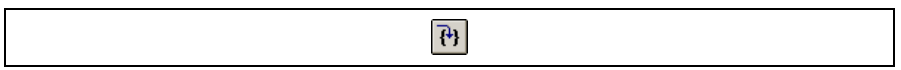


Figure 4.22 [Step In] Button

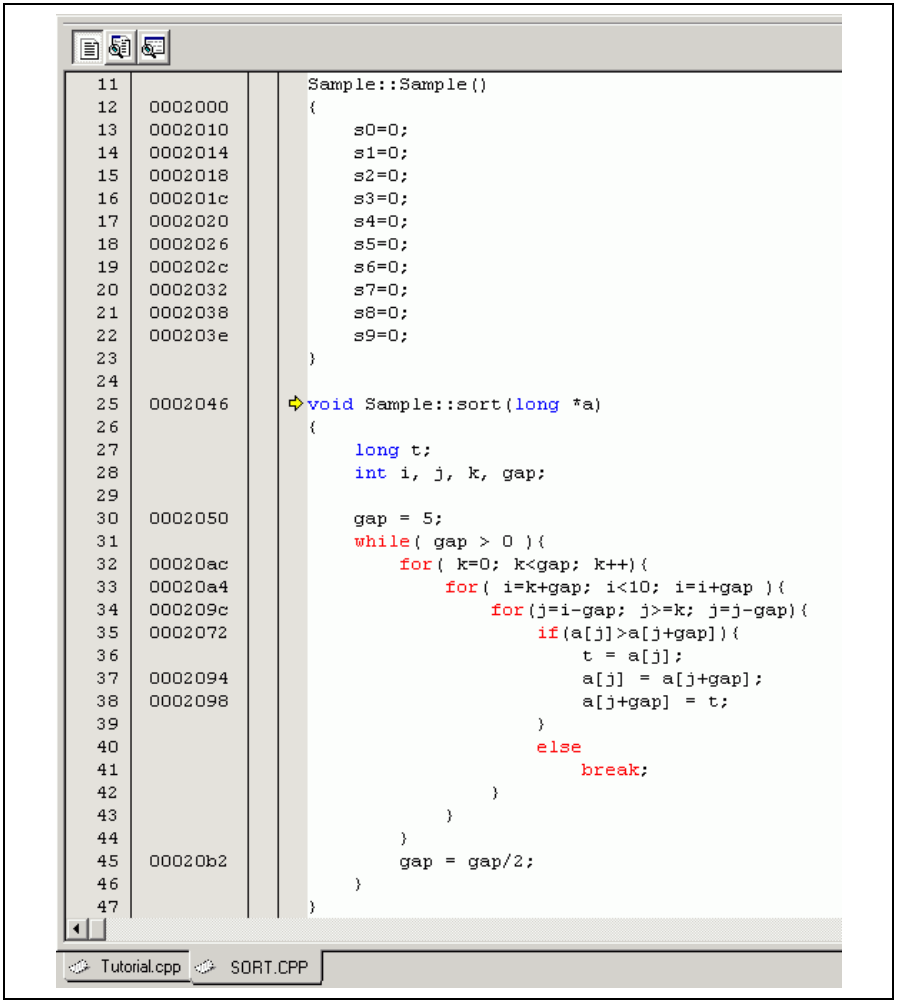


Figure 4.23 [Editor] Window (Step In)

- The highlighted line moves to the first statement of the `sort` function in the [Editor] window.

4.13.2 Executing the [Step Out] Command

The [Step Out] command steps out of the called function and stops at the next statement of the calling statement.

- To step out of the `sort` function, select [Step Out] from the [Debug] menu, or click the [Step Out] button in the toolbar.

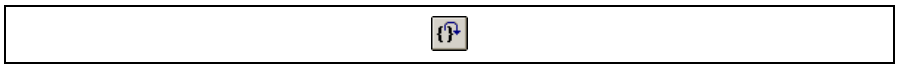


Figure 4.24 [Step Out] Button

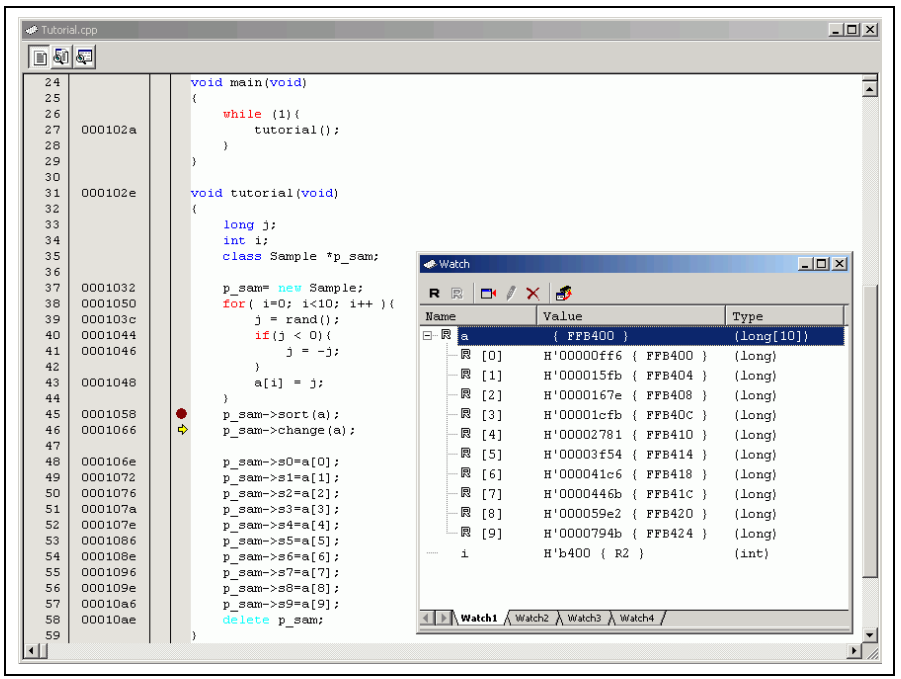


Figure 4.25 [High-performance Embedded Workshop] Window (Step Out)

The data of variable a displayed in the [Watch] window is sorted in the ascending order.

4.13.3 Executing the [Step Over] Command

The [Step Over] executes a function call in a single step and stops at the next statement of the main program.

- To step through all statements in the `change` function in a single step, select [Step Over] from the [Debug] menu, or click the [Step Over] button on the toolbar.

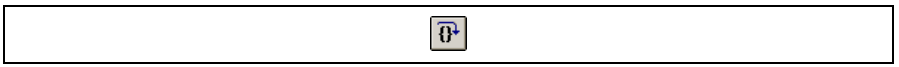


Figure 4.26 [Step Over] Button

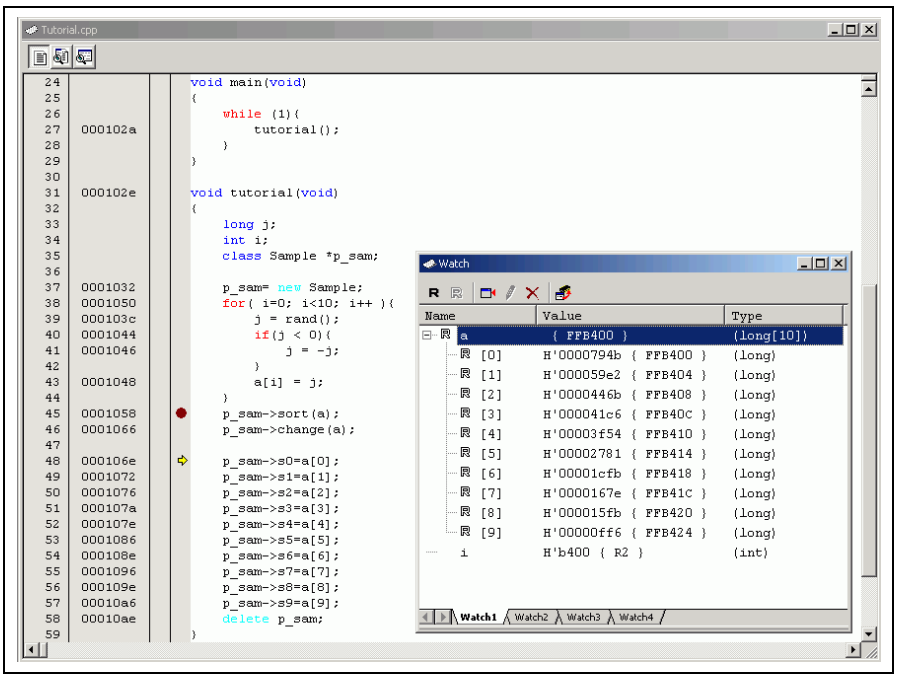


Figure 4.27 [High-performance Embedded Workshop] Window (Step Over)

The data of variable a displayed in the [Watch] window is sorted in the descending order.

4.14 Forced Breaking of Program Executions

The High-performance Embedded Workshop can force a break during the execution of a program.

- Cancel all the breakpoints.
- To execute the remaining sections of the `tutorial` function, select [Go] from the [Debug] menu or the [Go] button on the toolbar.

Figure 4.28 [Go] Button

- The program goes into an endless loop. To force a break during execution, select [Halt] from the [Debug] menu or the [Halt] button on the toolbar.

Figure 4.29 [Halt] Button

4.15 Resetting the Target MCU

Resetting the target MCU initializes the on-chip I/O registers and makes the program counter jump to the address set in the reset vector.

- To reset the target MCU, select [Reset CPU] from the [Debug] menu or the [Reset CPU] button on the toolbar.

Figure 4.30 [Reset CPU] Button

- To execute the user program immediately after a reset, select [Reset Go] from the [Debug] menu or the [Reset Go] button on the toolbar.

Figure 4.31 [Reset Go] Button

Note: This tutorial program is executable from the reset vector.

4.16 Break Function

The emulator provides break functions by software breaks, on-chip breaks, and on-emulator breaks. Software breakpoints, on-chip breakpoints, and on-emulator breakpoints can be set in the High-performance Embedded Workshop's [Event] window.

An overview and setting of the break function are described below.

4.16.1 Software Break Function

The emulator can set up to 255 software breakpoints.

- Select [Eventpoints] from the [Code] submenu of the [View] menu or click the [Eventpoints] toolbar button (). The [Event] window is displayed.
- Select the [Software] sheet.

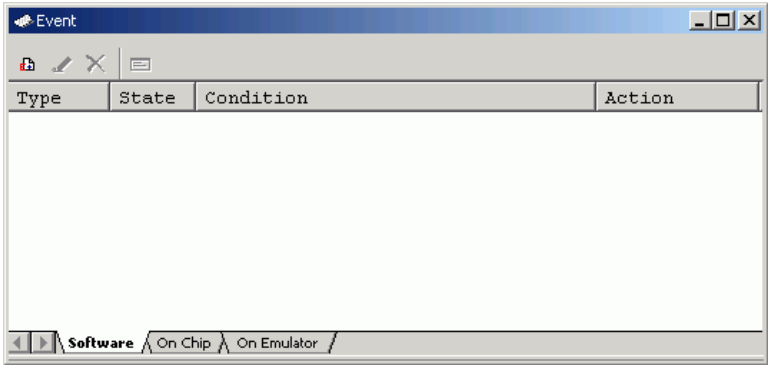


Figure 4.32 [Event] Window (Before Setting a Software Breakpoint)

- Click the [Event] window with the right-hand mouse button and select [Add…] from the popup menu.
- The [Breakpoint Properties] dialog box ([Software Break] page) is displayed.

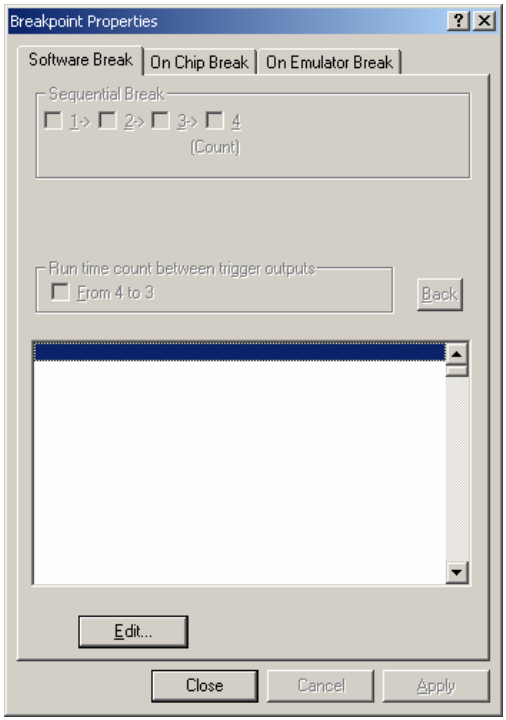


Figure 4.33 [Breakpoint Properties] Dialog Box

- Click the [Edit…] button to display the [Software Break] dialog box.

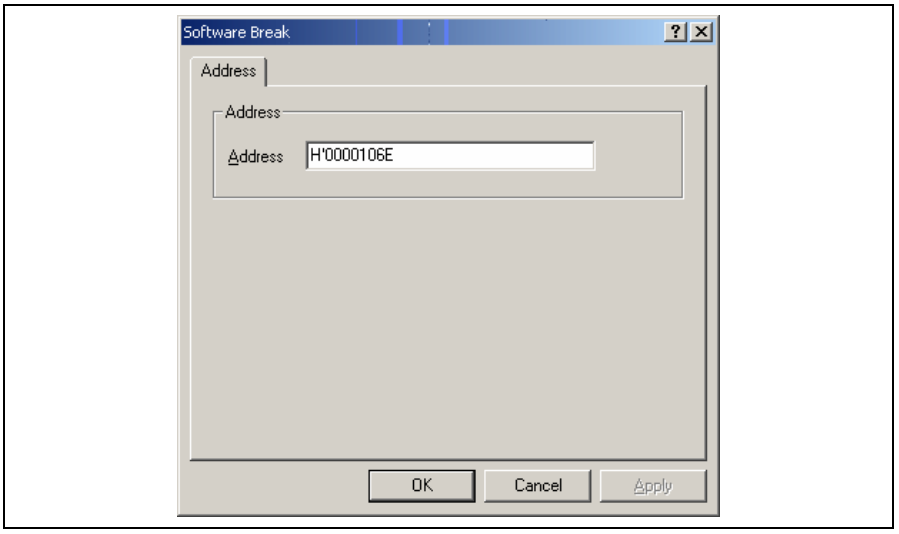


Figure 4.34 [Software Break] Dialog Box

- Use the [Editor] window to refer to the address on the line that has ‘p_sam->s0=a[0];’ within the `tutorial` function and enter this address in the [Address] edit box. In this example, enter ***H'0000106E***.

Note: This dialog box differs depending on the product. For the items of each product, refer to section 3, Debugging, or the online help.

- Click the [OK] button. Then click the [Close] button on the [Breakpoint Properties] dialog box.

The software breakpoint that has been set is displayed in the [Event] window.

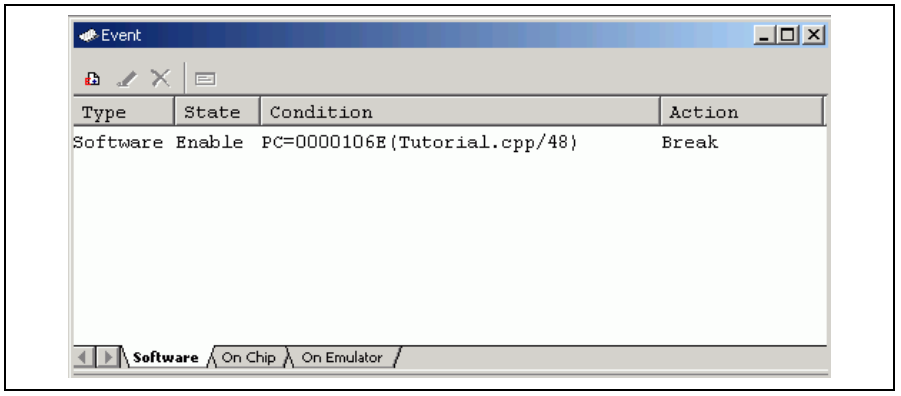


Figure 4.35 [Event] Window (Software Breakpoint Setting)

Note: The items that can be displayed in this window differ depending on the product. For the items that can be displayed, refer to section 3, Debugging, or the online help.

- Close the [Event] window.
- To stop the tutorial program at the breakpoint, select [Reset Go] from the [Debug] menu.

The program runs until it stops at the breakpoint that has been set.

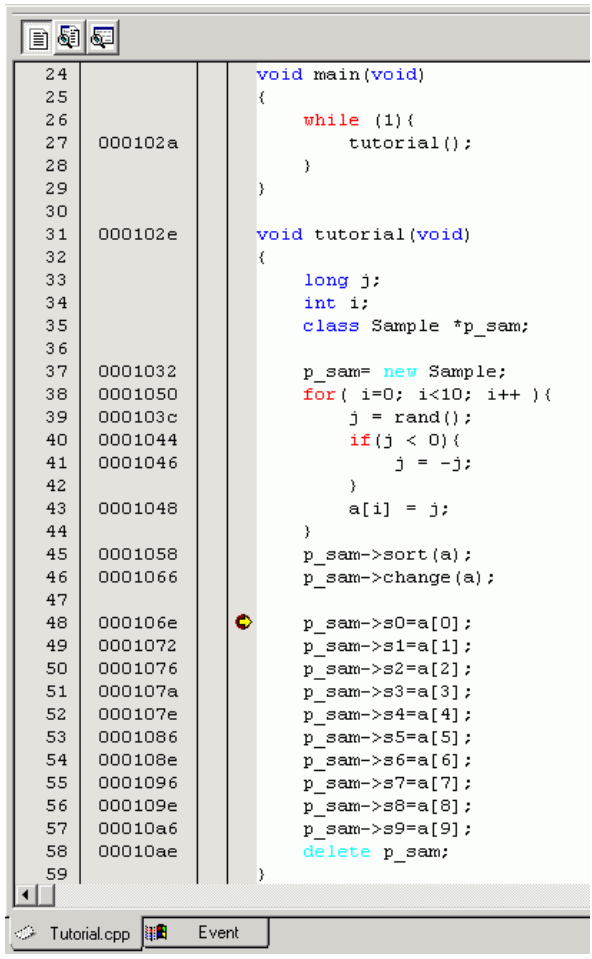


Figure 4.36 [Editor] Window at Execution Stop (Software Break)

The [Status] window displays the following contents:

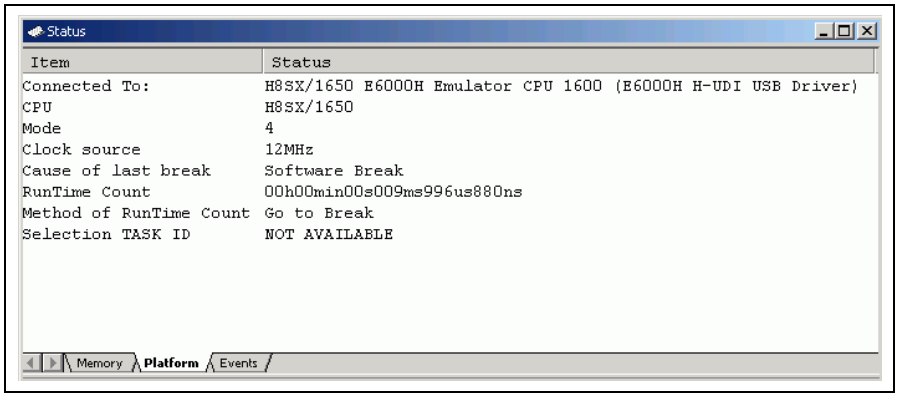


Figure 4.37 Displayed Contents of the [Status] Window (Software Break)

Note: The items that can be displayed in this window differ depending on the product. For the items that can be displayed, refer to section 3, Debugging, or the online help.

4.16.2 On-Chip Break Function

Setting of an on-chip breakpoint on channel 4 such that a break is triggered when the break condition has been satisfied five times is explained as an example of the use of on-chip breakpoints.

Note: The channels on which the satisfaction count can be specified differ depending on the product. For details on each product, refer to section 3, Debugging, or the online help.

- Select [Eventpoints] from the [Code] submenu of the [View] menu or click the [Eventpoints] toolbar button ( ). The [Event] window is displayed.
- The software breakpoint that has been previously set must be deleted. Click the [Software] sheet of the [Event] window with the right-hand mouse button and select [Delete All] from the popup menu to delete all the software breakpoints that have been set.
- Click the [On Chip] tab of the [Event] window.
- Click the [Event] window with the right-hand mouse button and select [Add…] from the popup menu.
- The [Breakpoint Properties] dialog box ([On Chip Break] page) is displayed.

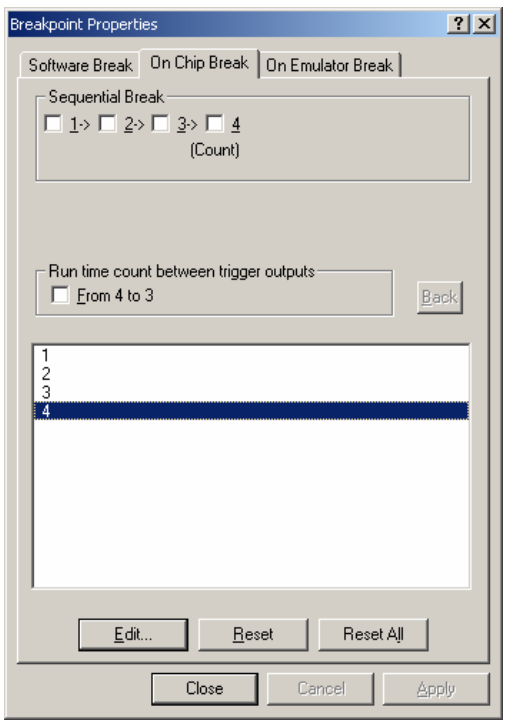


Figure 4.38 [Breakpoint Properties] Dialog Box ([On Chip Break] Page)

- Select [4] in the list box and click the [Edit…] button. The [On Chip Break Channel 4] dialog box is displayed.

- Make the following settings in the group boxes on the [Address] page:
  Uncheck the [Don't Care] checkbox.
  Then use the [Editor] window to refer to the address on the line that has 'a[i]=j;' within the `tutorial` function and enter this address in the [Address] edit box. In this example, enter ***H'0000105c***.
- Make the following settings in the boxes on the [Count] page:
  Uncheck the [Don't Care] checkbox.
  Enter ***D'5*** in the [Count] edit box.

Note: The content of this dialog box differs depending on the product. For details on each product, refer to section 3, Debugging, or the online help.

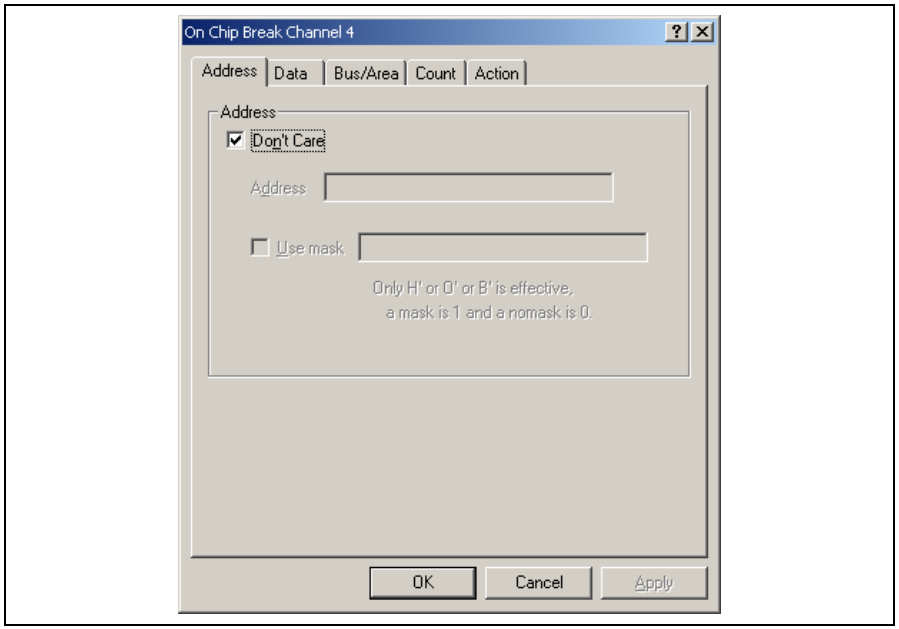


Figure 4.39 [On Chip Break Channel 4] Dialog Box

- Click the [OK] button. Then click the [Close] button on the [Breakpoint Properties] dialog box.

The on-chip breakpoint that has been set is displayed in the [Event] window.

Note: The items that can be displayed in this window differ depending on the product. For the items that can be displayed, refer to section 3, Debugging, or the online help.

Close the [Event] window. Then select [Reset Go] from the [Debug] menu to stop the tutorial program at on-chip breakpoints. The program runs and then stops at the breakpoint that has been set. The cause of a break can be checked in [Cause of last break] of the [Status] window.

Refer to the [Watch] window for the value of variable `i`. The value is 4, indicating that the break occurred after the condition had been satisfied five times.

Then delete the on-chip breakpoint. Clicking the right-hand mouse button on the [Event] window displays a popup menu. Select [Delete All] from this menu to delete all the on-chip breakpoints.

4.17 Trace Functions

The trace functions of the emulator use the realtime trace buffer, which can store the information of up to 128-k bus cycles. The content of this buffer, which is constantly updated during execution, is displayed in the [Trace] window.

Select [Trace] from the [Code] submenu of the [View] menu or click the [Trace] toolbar button ( ) to display the [Trace] window.

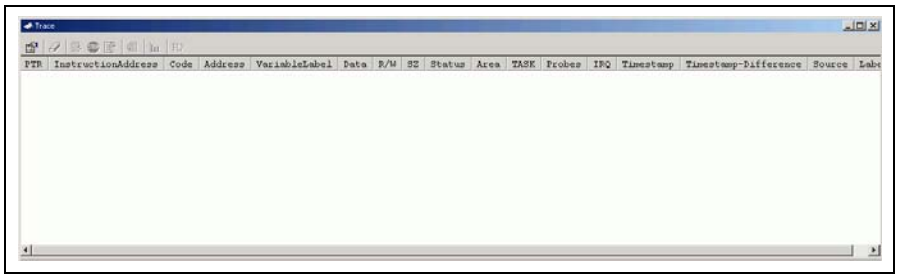


Figure 4.40 [Trace] Window

When trace information is displayed in the [Trace] window, clicking the right-hand mouse button on the [Trace] window displays a popup menu. Select [Clear] from this menu to clear the trace information.

The following sections give an overview of the trace functions and the settings.

4.17.1 Displaying Trace Information by the Free Trace Function

The free trace function allows continuous acquisition of trace information from the start of user program execution to the occurrence of a break.

(1) All break conditions must be deleted. Clicking the right-hand mouse button on the [Trace] window displays a popup menu. Select [Acquisition…] from this menu to display the [Trace Acquisition Properties] dialog box. Ensure that [Free Trace] is checked and then click the [Close] button.

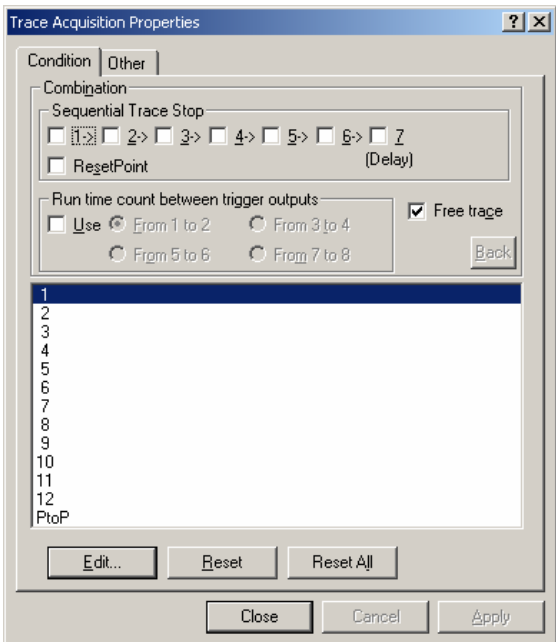


Figure 4.41 [Trace Acquisition Properties] Dialog Box (Free Trace)

(2) Set a software breakpoint at the address on the line that has ‘p_sam->s0=a[0];’ within the `tutorial` function (refer to section 4.16.1, Software Break Function).

(3) Select [Reset Go] from the [Debug] menu. Execution stops when the break condition is satisfied, and the [Trace] window then displays the trace information.

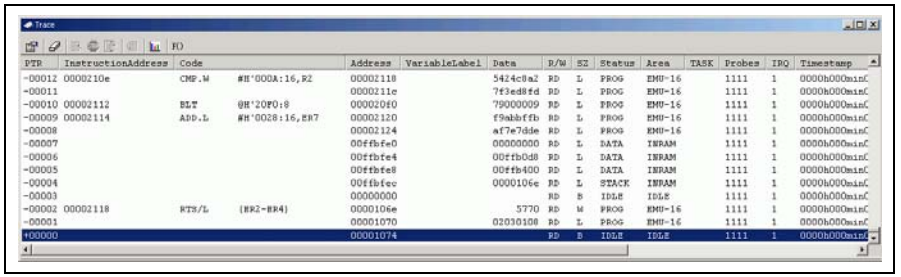


Figure 4.42 [Trace] Window (Free Trace)

4.17.2 Displaying Trace Information by the Trace Stop Function

While the trace stop function is in use, acquisition of trace information stops when a specified condition is satisfied. The user can check the program flow by the trace information without breaking the user program execution.

(1) Delete all the break conditions that have been set. Uncheck [Free Trace] on the [Condition] page of the [Trace Acquisition Properties] dialog box (otherwise, the free trace mode will be selected).

(2) Select [1] from the list box on the [Condition] page of the [Trace Acquisition Properties] dialog box and then click [Edit…]. The [Trace Acquisition Condition Channel 1] dialog box is displayed. Select the [Trace Stop] radio button in the [After Condition Match] group box on the [Action] page.

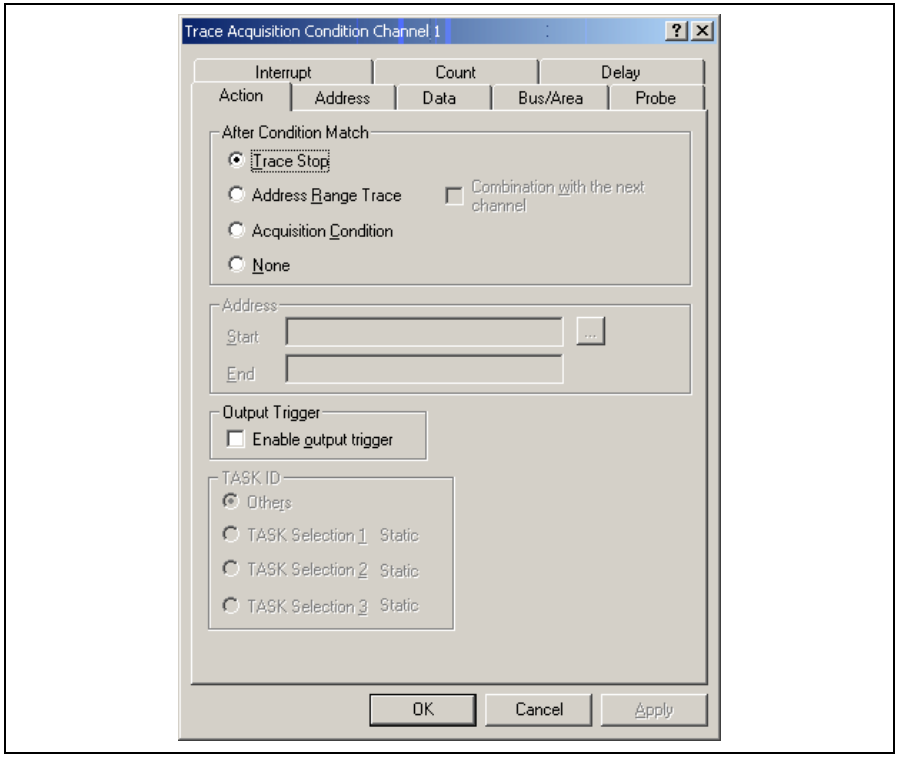


Figure 4.43 [Trace Acquisition Condition Channel 1] Dialog Box (Trace Stop)

(3) An address must be set as the condition. Uncheck [Don't Care] on the [Address] page of the [Trace Acquisition Condition Channel 1] dialog box. Then use the [Editor] window to refer to the address on the line that includes 'a[i]=j;' within the `tutorial` function and enter this address in the [Start] edit box. In this example, enter ***H'00001048***. This completes the setting of the address. Click the [OK] button to close the [Trace Acquisition Condition Channel 1] dialog box.

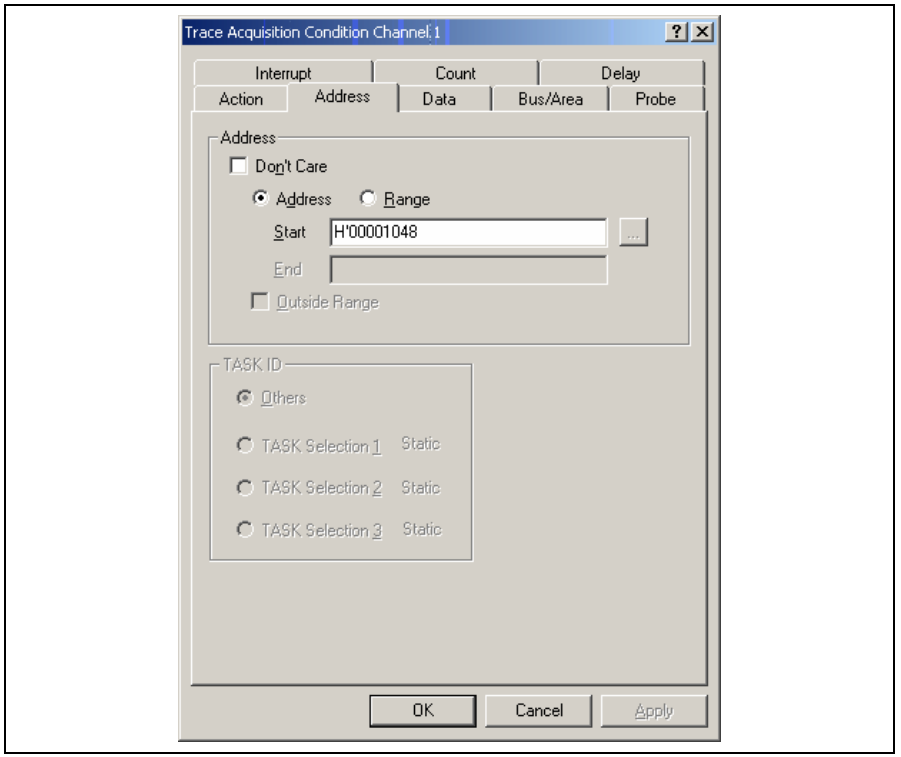


Figure 4.44 [Trace Acquisition Condition Channel 1] Dialog Box ([Address] Page)

(4) Items that have been set are displayed in the list box on the [Condition] page of the [Trace Acquisition Properties] dialog box. Click the [Close] button on this dialog box.

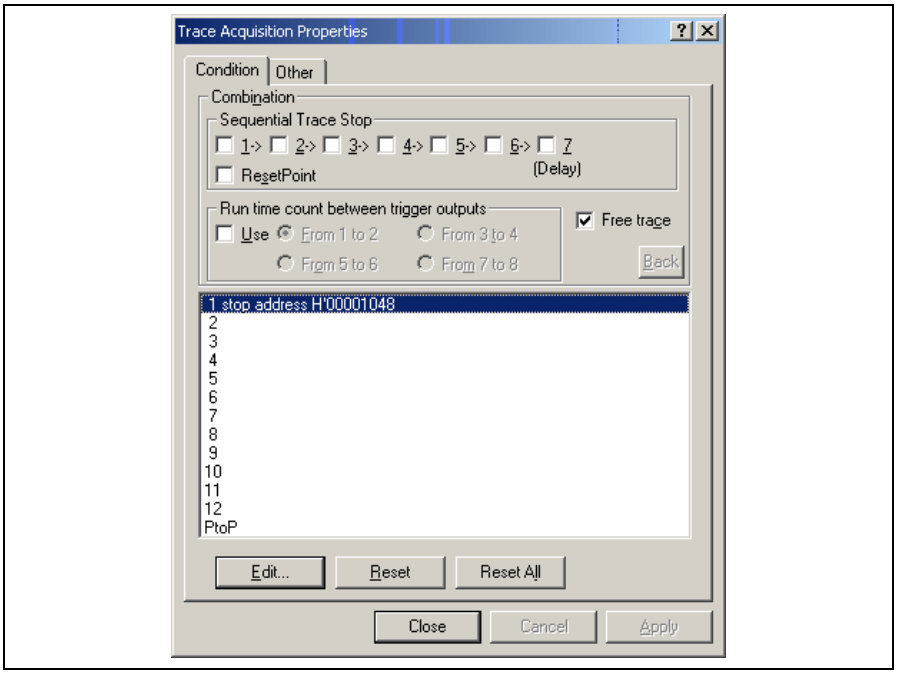


Figure 4.45 [Trace Acquisition Properties] Dialog Box (Trace Stop)

(5) Select [Reset Go] from the [Debug] menu. The trace condition is satisfied, and the [Trace] window then displays the following contents.

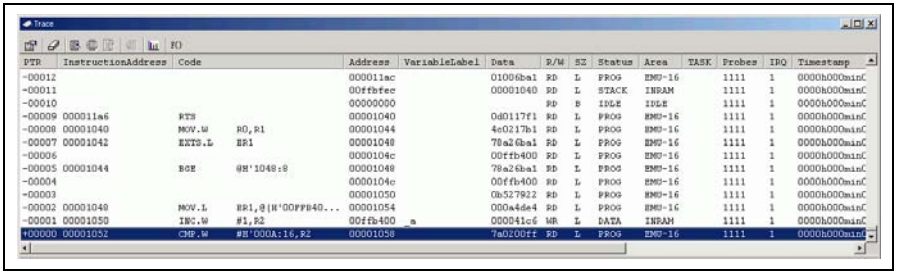


Figure 4.46 [Trace] Window (Trace Stop)

4.17.3 Displaying Trace Information by the Conditional Trace Function

The conditional trace function only acquires trace information at the address where a specified condition has been satisfied. This is useful for analyzing a program focused on reading from or writing to a specific address (e.g. a global variable or memory mapped I/O).

(1) If the user program is running, select [Halt Program] from the [Debug] menu to halt the program.

(2) Delete all the break conditions that have been set. Uncheck [Free Trace] on the [Condition] page of the [Trace Acquisition Properties] dialog box (otherwise, the free trace mode will be selected).

(3) Select [1] from the list box on the [Condition] page of the [Trace Acquisition Properties] dialog box and then click [Edit…]. The [Trace Acquisition Condition Channel 1] dialog box is displayed. Select the [Acquisition Condition] radio button in the [After Condition Match] group box on the [Action] page.

(4) An address must be set as the condition. Uncheck [Don't Care] on the [Address] page of the [Trace Acquisition Condition Channel 1] dialog box. Then use the [Watch] window to refer to the address on the line that includes 'a[0]' and enter this address in the [Start] edit box. In this example, enter ***H'00FFB400***. This completes the setting of an address. Click the [OK] button to close the [Trace Acquisition Condition Channel 1] dialog box.

(5) Items that have been set are displayed in the list box on the [Condition] page of the [Trace Acquisition Properties] dialog box. Click the [Close] button on this dialog box.

(6) Set a software breakpoint at the address on the line that has 'delete p_sam;' within the `tutorial` function (***H'000010AE***. in this example) (for details, refer to section 4.16.1, Software Break Function).

(7) Select [Reset Go] from the [Debug] menu. Execution stops when the break condition is satisfied, and the [Trace] window then displays the following contents.

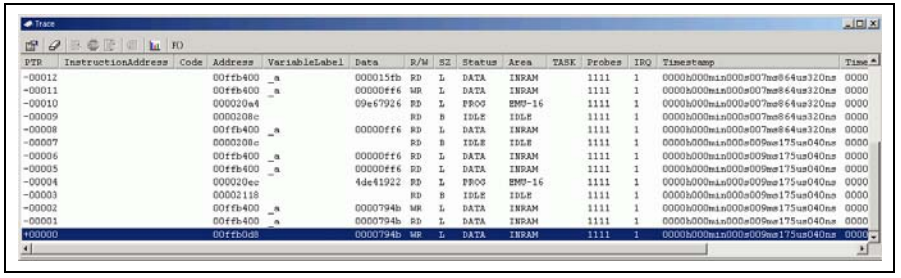

Trace
PTP InstructionAddress Code Address VariableLabel Data R/W SZ Status Area TASK Probes IRQ Timestamp Time
-00012 00ffb400 _a 000015fb RD L DATA INRAM 1111 1
-00011 00ffb400 _a 00000ff6 WR L DATA INRAM 1111 1
-00010 000020a4 09e67926 RD L PROG EMU-16 1111 1
-00009 0000208c RD B IDLE IDLE 1111 1
-00008 00ffb400 _a 00000ff6 RD L DATA INRAM 1111 1
-00007 0000208c RD B IDLE IDLE 1111 1
-00006 00ffb400 _a 00000ff6 RD L DATA INRAM 1111 1
-00005 00ffb400 _a 00000ff6 RD L DATA INRAM 1111 1
-00004 000020ec 4de61922 RD L PROG EMU-16 1111 1
-00003 00002118 RD B IDLE IDLE 1111 1
-00002 00ffb400 _a 0000794b WR L DATA INRAM 1111 1
-00001 00ffb400 _a 0000794b RD L DATA INRAM 1111 1
+00000 00ffb0d8 0000794b WR L DATA INRAM 1111 1

Figure 4.47 [Trace] Window (Conditional Trace)

4.17.4 Statistics

The number of times the on-chip RAM has been written to can be included in the acquired trace information.

(1) Delete all the break conditions that have been set. Click [Reset All] on the [Condition] page of the [Trace Acquisition Properties] dialog box to cancel trace conditions. Check [Free Trace] on the [Condition] page of the [Trace Acquisition Properties] dialog box.

(2) Make the setting so that a break occurs at the address on the line that has 'p_sam->s0=a[0];' within the `tutorial` function (***H'0000106E*** in this example) (for details on, refer to section 4.16.1, Software Break Function).

(3) Select [Reset Go] from the [Debug] menu. Execution stops when the break condition is satisfied, and the [Trace] window then displays the trace information.

(4) Select [Statistic…] from the popup menu that is displayed when you click the right-hand mouse button on the [Trace] window. A message box appears, indicating that the trace data is being loaded, and the [Statistic] dialog box will be displayed.

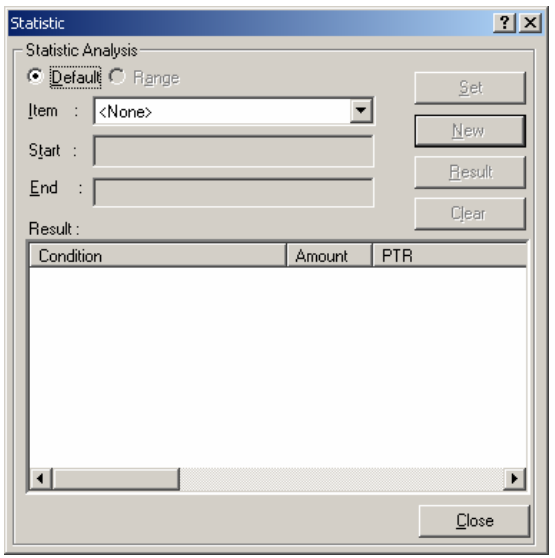


Figure 4.48 [Statistic] Dialog Box

(5) Select [R/W] in the [Item] combo box and enter ***WR*** in the [Start] edit box. Then, click the [New] button.
"R/W=WR" will be displayed in the [Condition] column of the [Result] list box.

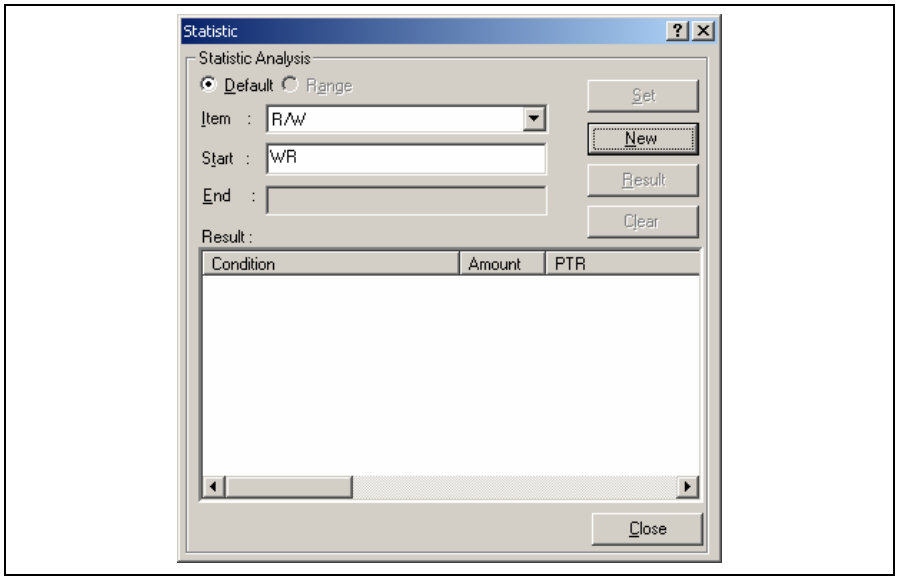


Figure 4.49 [Statistic] Dialog Box (New Condition)

(6) Then, select [Area] from the [Item] combo box and enter ***INRAM*** in the [Start] edit box. Then, click the [Add] button; the new condition is now added to the "R/W=WR" display in the [Condition] column of the [Result] list box, so that it now shows "R/W=WR & Area=INRAM". This completes the setting of the conditions.

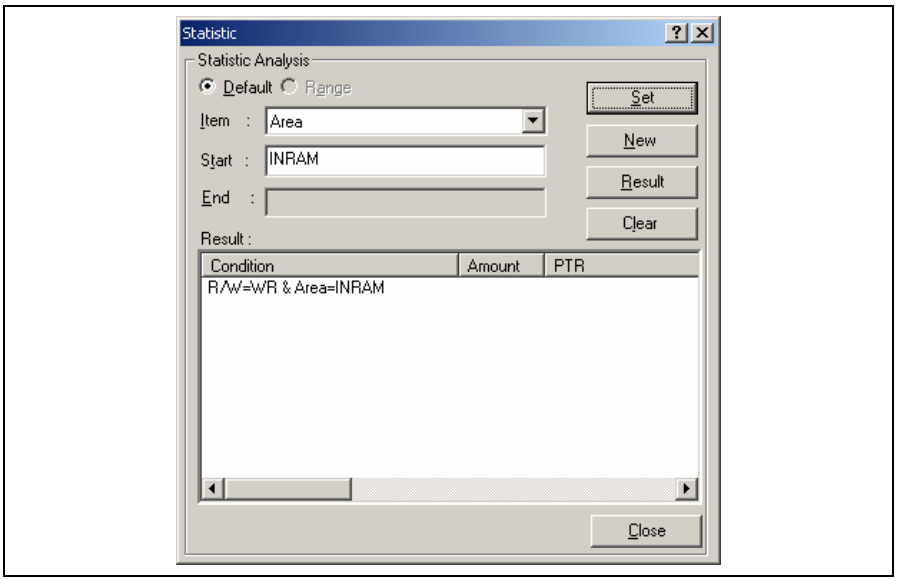


Figure 4.50 [Statistic] Dialog Box (Condition Added)

(7) To start statistical analysis of the specified condition, press the [Result] button. The number of write operations that satisfies the conditions and the PTR values will be displayed.

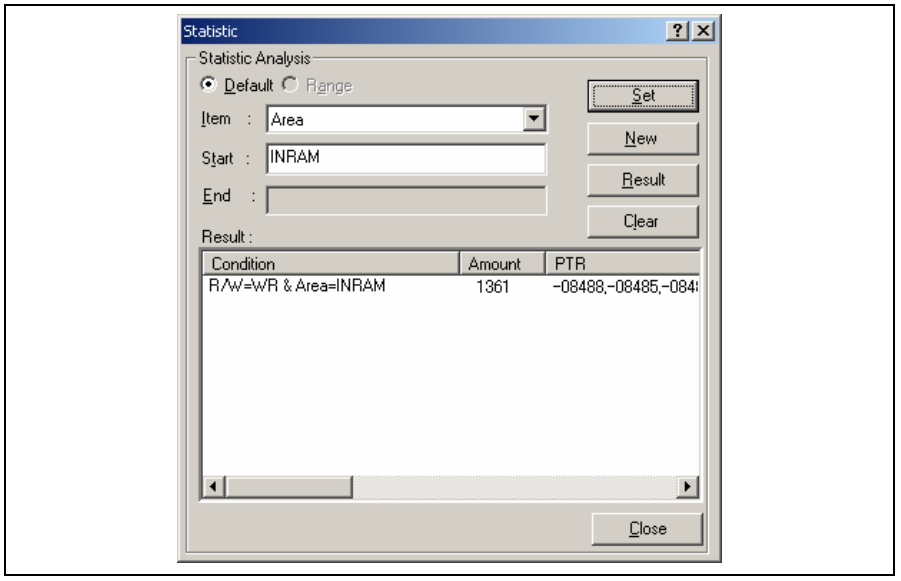


Figure 4.51 [Statistic] Dialog Box (Result of Analysis)

(8) Click the [Close] button to close the [Statistic] dialog box.

(9) Delete the event points that have been set and clear the trace information. Clicking the right-hand mouse button on the [Event] window displays a popup menu. Select [Delete All] from this menu to delete all the event points that have been set. Clicking the right-hand mouse button on the [Trace] window displays a further popup menu. Select [Clear] from this menu to clear the trace information.

4.17.5 Function Calls

This mechanism is only used to collect trace information on the function calls.

(1) Make the setting so that a break occurs at the address on the line that has 'p_sam->s0=a[0];' within the `tutorial` function (***H'0000106e*** in this example) (for details, refer to section 4.16.1, Software Break Function).

(2) Select [Reset Go] from the [Debug] menu. Execution stops when the break condition is satisfied, and the [Trace] window then displays the trace information.

(3) Select [Function Call…] from the popup menu displayed by clicking the right-hand mouse button on the [Trace] window. The [Function Call Display] dialog box will be displayed.

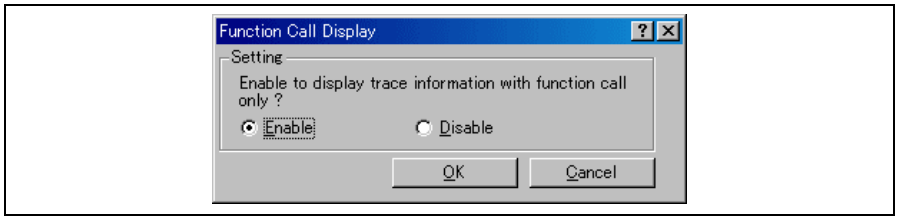


Figure 4.52 [Function Call Display] Dialog Box

(4) Click the [Enable] radio button and then the [OK] button. Only the information on function calls is now displayed in the [Trace] window.

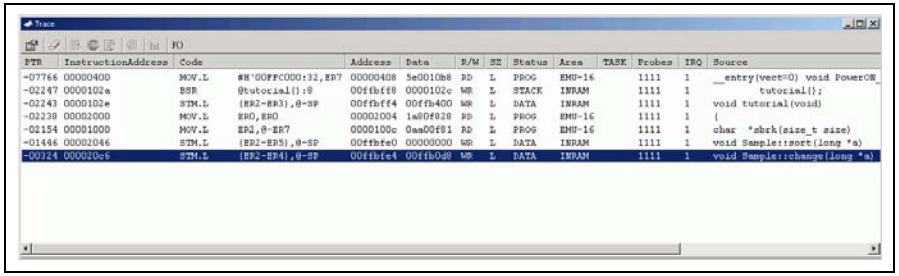

Trace
PTR InstructionAddress Code Address Data B/W SZ Status Area TASK Probes IRQ Source
-07766 00000400 MOV.L #H'00FFC000:32,ER7 00000408 5e0010b8 RD L PROG EMU-16 1111 1 __entry(vect=0) void PowerON_
-02247 0000102a BSR @tutorial():8 00ffbff8 0000102c WR L STACK INRAM 1111 1 tutorial();
-02243 0000102e STM.L (ER2-ER3),@-SP 00ffbff4 00ffb400 WR L DATA INRAM 1111 1 void tutorial(void)
-02238 00002000 MOV.L ER0,ER0 00002004 1a80f820 RD L PROG EMU-16 1111 1 {
-02154 00001000 MOV.L ER2,@-ER7 0000100c 0aa00f81 RD L PROG EMU-16 1111 1 char *sbrk(size_t size)
-01446 00002046 STM.L (ER2-ER5),@-SP 00ffbfe0 00000000 WR L DATA INRAM 1111 1 void Sample::sort(long *a)
-00324 000020c6 STM.L (ER2-ER4),@-SP 00ffbfe4 00ffb0d8 WR L DATA INRAM 1111 1 void Sample::change(long *a)

Figure 4.53 [Trace] Window (Function Calls)

(5) To return the display in the [Trace] window to its previous state, follow the procedure in (3) to display the [Function Call Display] dialog box. Click the [Disable] button and then the [OK] button.

(6) Delete the event points that have been set and clear the trace information. Clicking the right-hand mouse button on the [Event] window displays a popup menu. Select [Delete All] from this menu to delete all the event points that have been set. Clicking the right-hand mouse button on the [Trace] window displays a further popup menu. Select [Clear] from this menu to clear the trace information.

4.18 Stack Trace Function

The emulator uses the information on the stack to display the function call history.

Notes: 1. This function can be used only when the load module that has the Dwarf2-type debugging information is loaded. Such load modules are supported in H8S, H8/300 C/C++ compiler V4.0 or later.

2. For details on the stack trace function, refer to the online help.

- Double-click the [S/W Breakpoints] column in the `sort` function and set a software breakpoint.

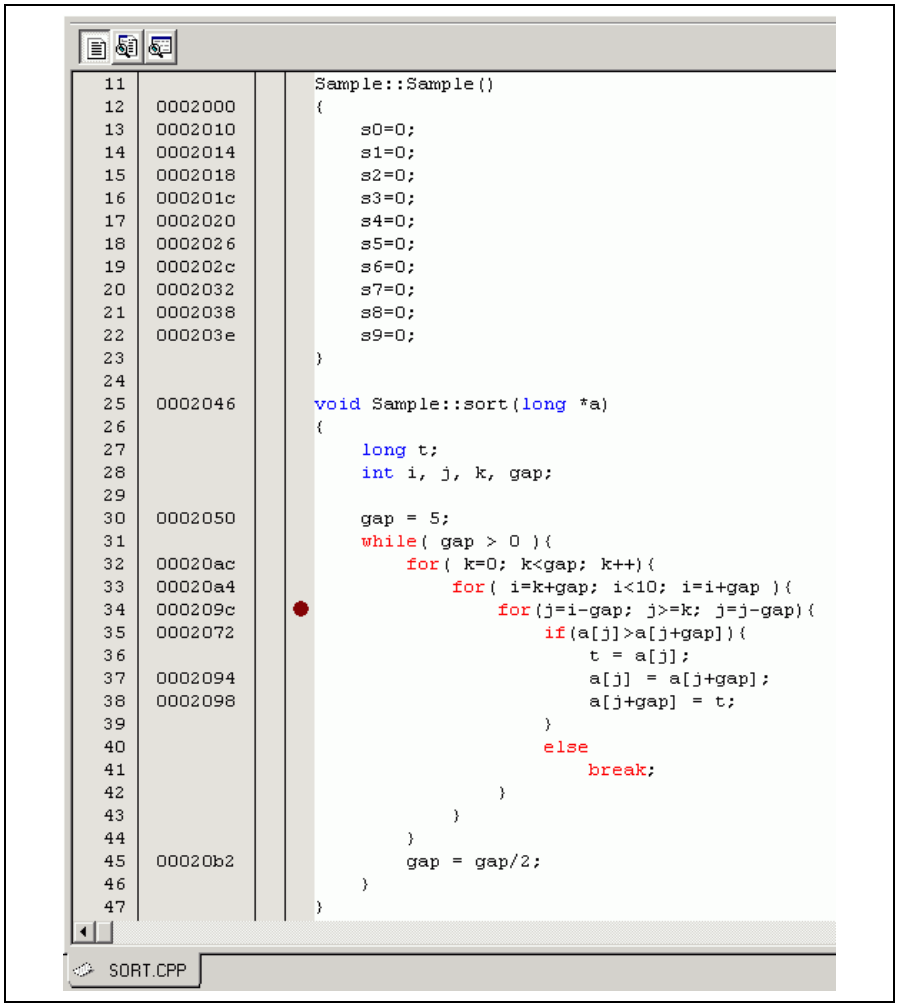


Figure 4.54 [Editor] Window (Software Breakpoint Setting)

- Select [Reset Go] from the [Debug] menu.
- After the break in program execution, select [Stack Trace] from the [Code] submenu of the [View] menu to open the [Stack Trace] window.

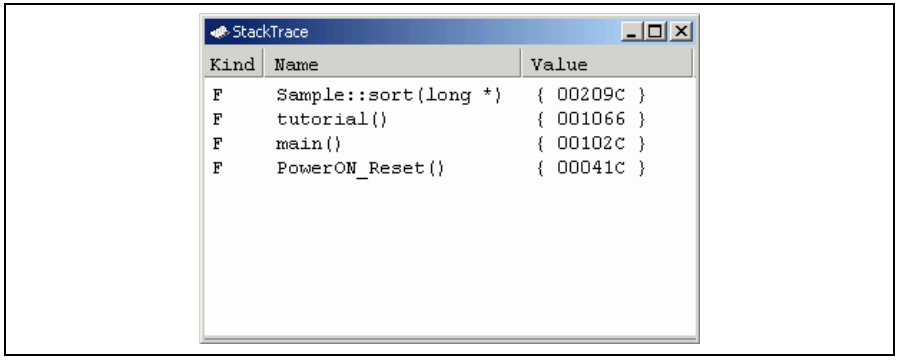


Figure 4.55 [Stack Trace] Window

Figure 4.55 shows that the position of the program counter is currently at the selected line of the `sort()` function, and that the `sort()` function is called from the `tutorial()` function.

To delete the software breakpoint, double-click the [S/W Breakpoints] column in the `sort` function again.

4.19 Performance Analysis Function

Performance analysis by the emulator is available in the following modes:

- Time Of Specified Range Measurement
- Start Point To End Point Measurement
- Start Range To End Range Measurement
- Access Count Of Specified Range Measurement
- Called Count Of Specified Range Measurement

In this tutorial, we describe the Time Of Specified Range Measurement.

4.19.1 Time Of Specified Range Measurement

(1) Select [Performance Analysis] from the [Performance] submenu of the [View] menu to display the [Select Performance Analysis Type] dialog box.

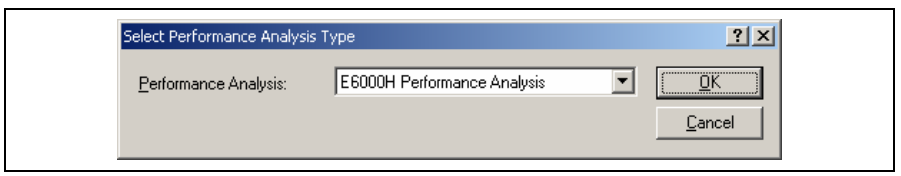


Figure 4.56 [Select Performance Analysis Type] Dialog Box

(2) Select “E6000H Performance Analysis” from the [Performance Analysis] combo box in the [Select Performance Analysis Type] dialog box and click the [OK] button. The [Performance Analysis] window will be displayed.

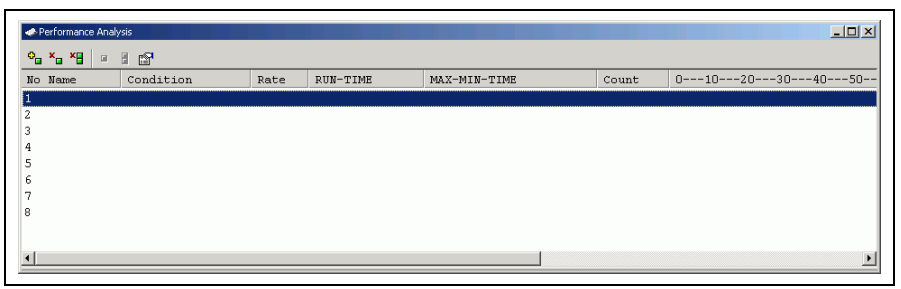


Figure 4.57 [Performance Analysis] Window

(3) Select the line in the [Performance Analysis] window that has `1` in its [No] column and click the right-hand mouse button to display a popup menu. Select [Set…] from this popup menu to display the [Performance Analysis Properties] dialog box.

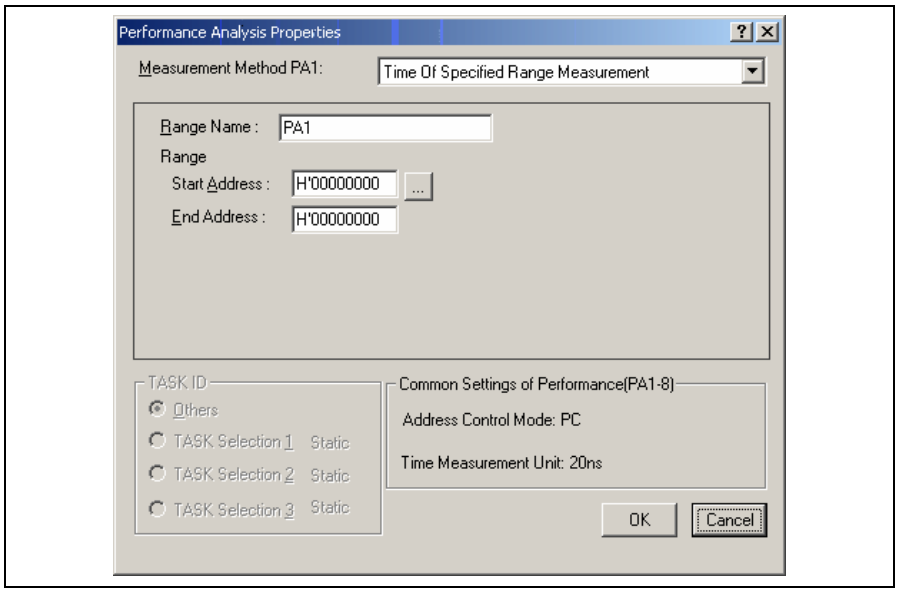


Figure 4.58 [Performance Analysis Properties] Dialog Box

(4)Select `Time Of Specified Range Measurement` from the [Measurement Method PA1] combo box.

(5) The parameter settings are as follows.

- Enter ***`sort`*** in the [Range Name] edit box.
- Click the […] button on the right of the [Start Address] edit box to display the [Input Function Range] dialog box. Enter the function name ***`sort`*** in the [Function] edit box in this dialog box and then click the [OK] button. The addresses for the function `Sample::sort(long*)` will now be set in the [Start Address] and [End Address] edit boxes.

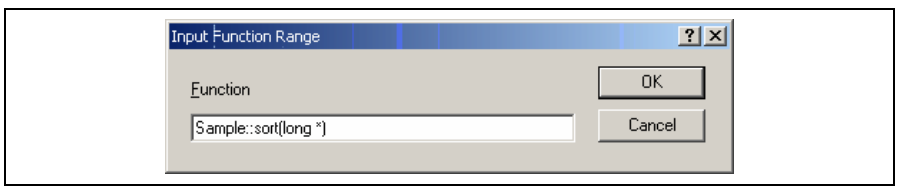


Figure 4.59 [Input Function Range] Dialog Box

Note: The addresses figured out in the [Input Function Range] dialog box are just for reference. In some cases, the end address of a function may be incorrect. Check the last instruction of the function in the [Disassembly] window to correct the value set in [End Address] so that it will be the address of the last instruction (in general, the last instruction of a function is a RTS instruction). A label name or an expression can be entered instead of an address value in boxes where an address should be entered.

(6) Click the [OK] button to display the contents that has been set for line 1 of the [No] column in the [Performance Analysis] window. This completes the settings for measuring the time within the specified range.

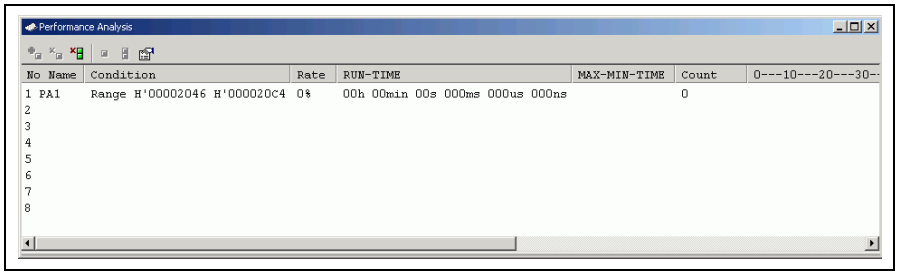


Figure 4.60 [Performance Analysis] Dialog Box (Setting Completed)

(7) Set a software breakpoint at the address on the line that has 'p_sam->change(a);' within the `tutorial` function (***H'00001066*** in this example). Refer to section 4.16.1, Software Break Function.

(8) Select [Reset Go] from the [Debug] menu. Execution stops when the break condition is satisfied, and the [Performance Analysis] window then displays the information shown below. The value shown in the [Count] column is 1, which indicates that the `sort` function has been executed once, and the execution time is also displayed. In this tutorial, the minimum unit for time measurement is defined as 20 ns. This value can be changed in the [Configuration Properties] dialog box.

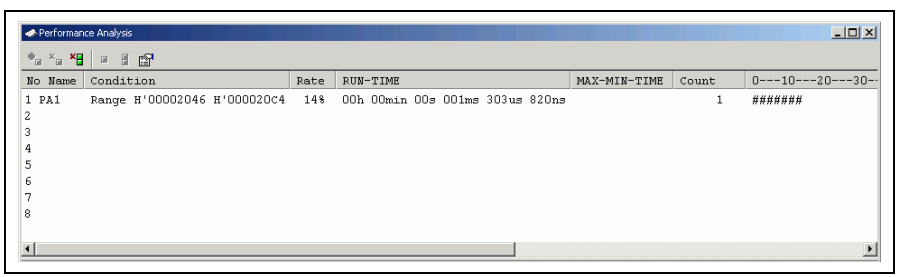


Figure 4.61 [Performance Analysis] Dialog Box (Displaying the Result)

(9) Delete the settings for performance analysis and delete the event points. Click the right-hand mouse button on the [Performance Analysis] window to display a popup menu. Select [Reset All] from this popup menu to clear all of the settings. Clicking the right-hand mouse button on the [Event] window also displays a popup menu. Select [Delete All] from this popup menu to delete all the event points that have been set.

4.20 Profiling Function

The profiling function allows the user measure the performance for each of the functions.

(1) Select [Profile] from the [View] menu to open the [Profile] window.

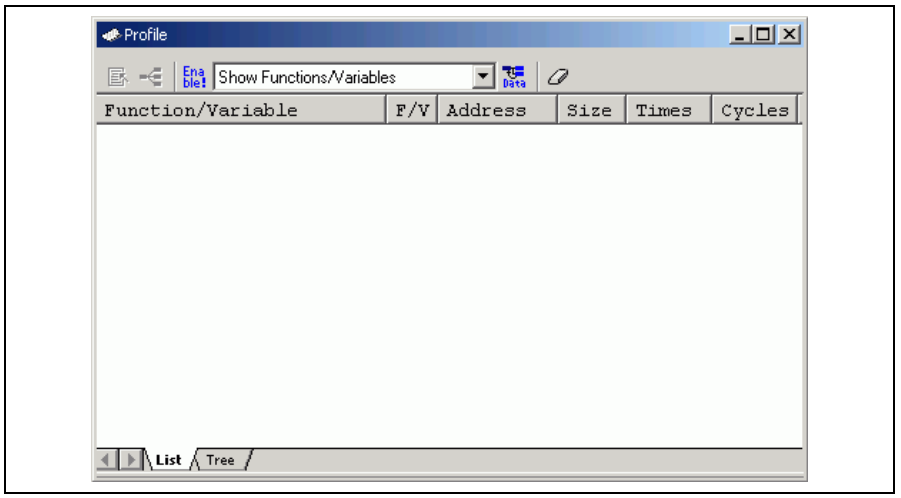


Figure 4.62 [Profile] Window ([List] Sheet)

(2) To enable the profiling function, click the right-hand mouse button on the [Profile] window to show the popup menu and select [Enable Profiler].

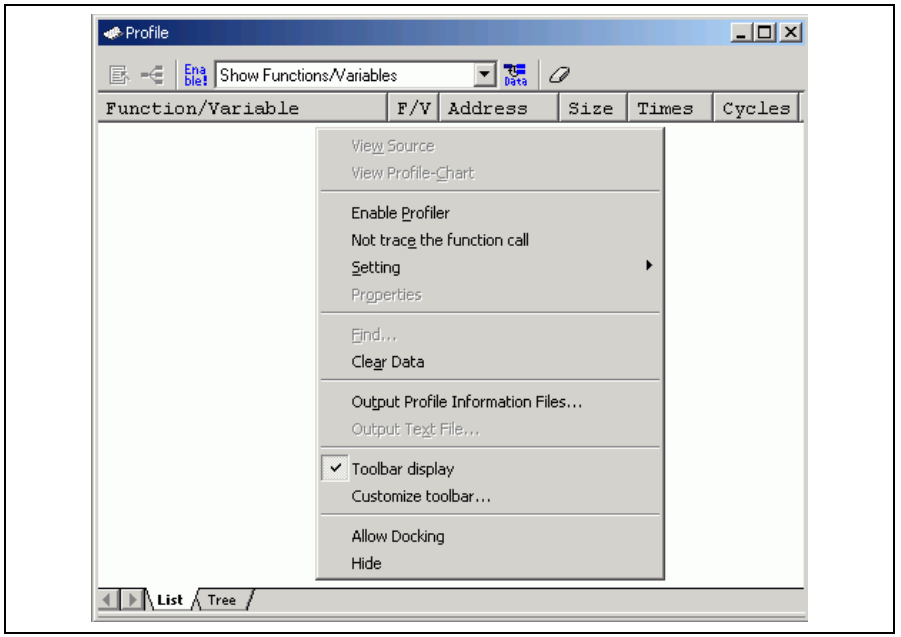


Figure 4.63 Selecting [Enable Profiler]

(3) Set an on-chip breakpoint by an address condition at the line which includes "`delete p_sam;`" in the `tutorial` function (see section 4.16.2, On-Chip Break Function).

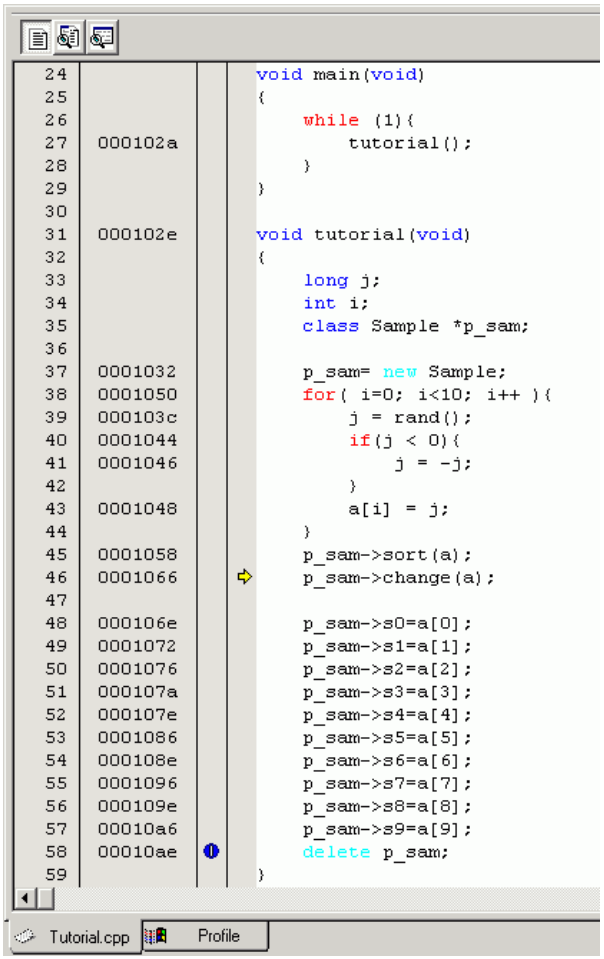


Figure 4.64 [Editor] Window (Setting an On-Chip Breakpoint)

(4) To use the profiling function for measurement, select [Reset Go] from the [Debug] menu.

(5) The [Profile] window is shown below.

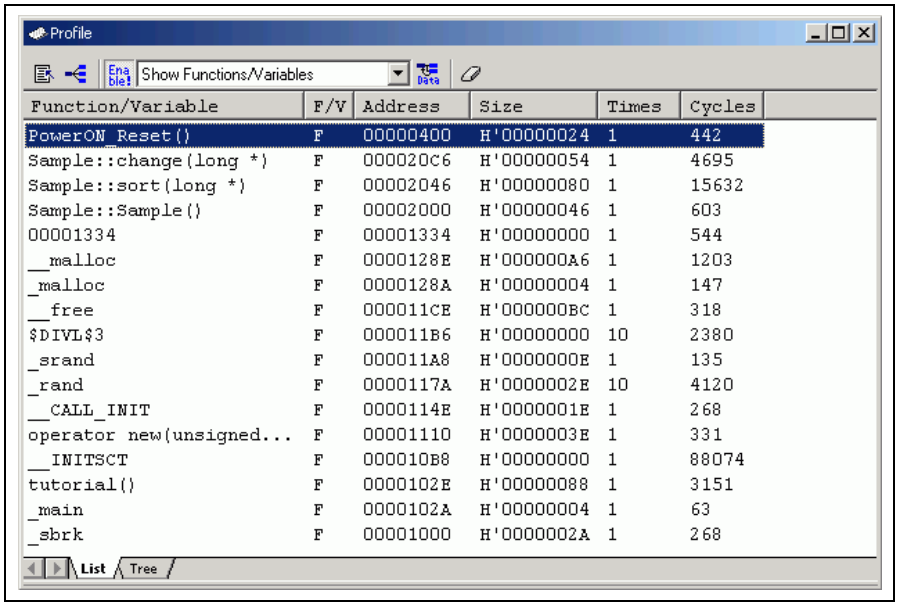


Figure 4.65 [Profile] Window ([List] Sheet)

(6) Click the [Tree] tab on the [Profile] window to display the [Tree] sheet.

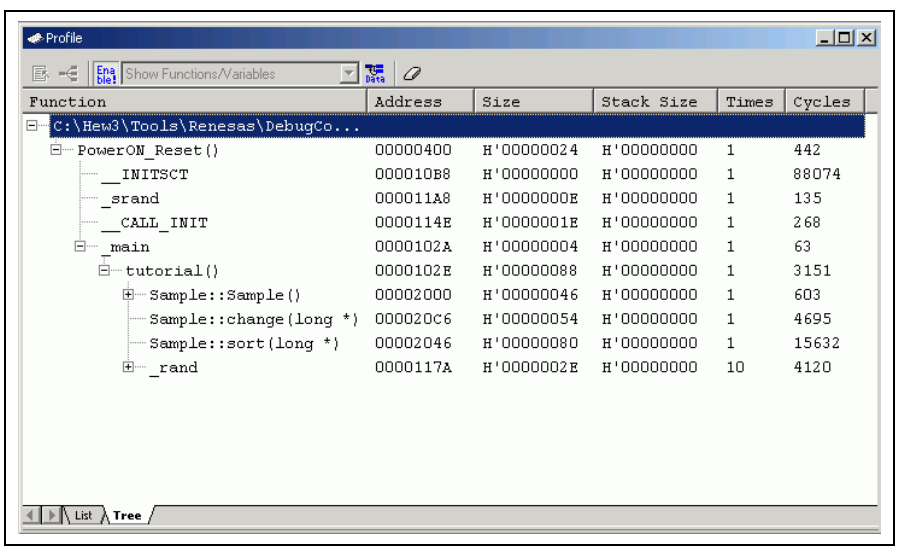


Figure 4.66 [Profile] Window ([Tree] Sheet)

(7) Click the right-hand mouse button on the [Profile] window and select [View Profile-Chart] to open the [Profile-Chart] window.

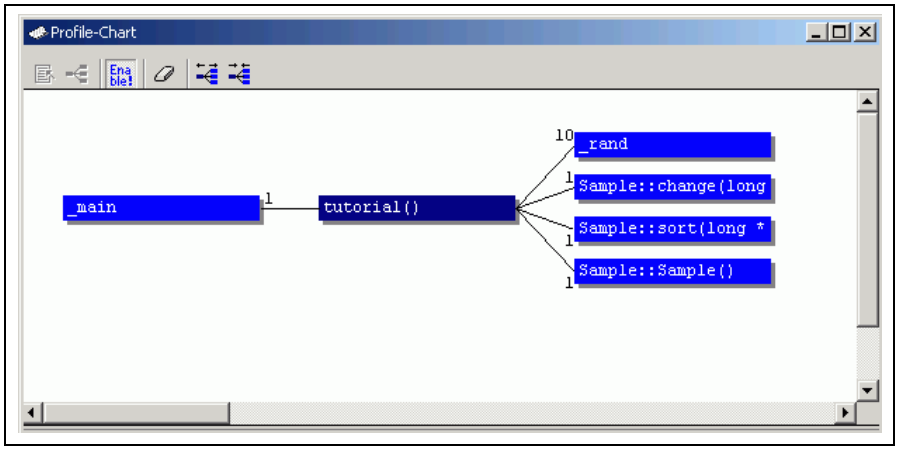


Figure 4.67 [Profile-Chart] Window

(8) To disable the profiling function, uncheck [Enable Profiler] in the popup menu opened by clicking the right-hand mouse button on the [Profile] window. Delete all the break conditions that have been set.

4.21 Monitor Function

The emulator allows monitoring of the contents of specified addresses in memory during execution of the user program. In this example, we monitor the content of the address range where variable `a` of the `tutorial` function is stored.

(1) Select the [CPU] submenu from the [View] menu. Selecting [Monitor Setting…] from the [Monitor] submenu displays the [Monitor Setting] dialog box.

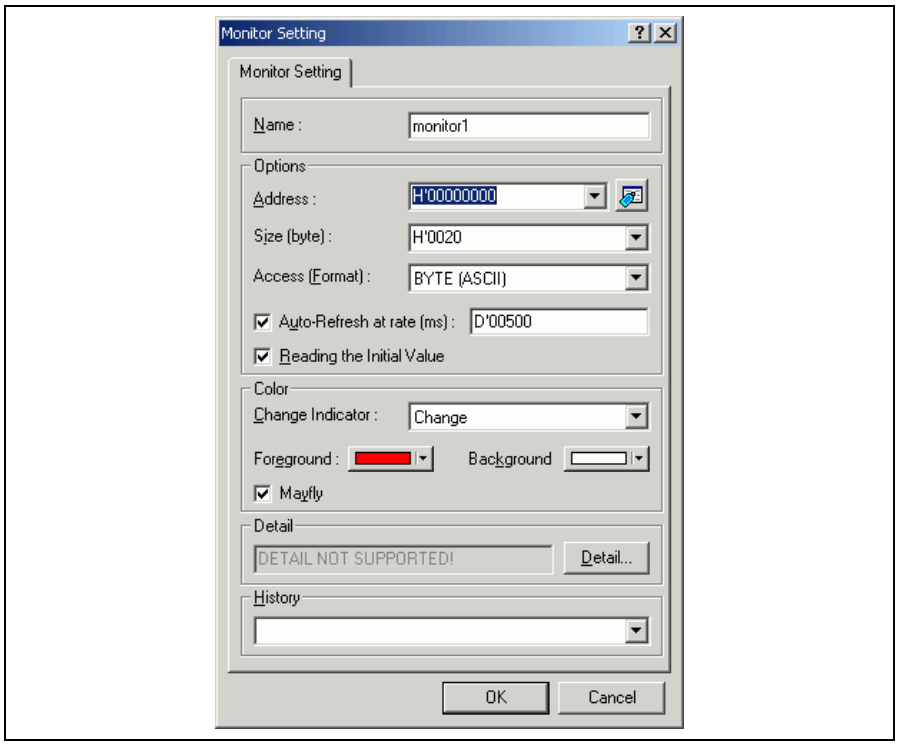


Figure 4.68 [Monitor Setting] Dialog Box

(2) Set the items in the [Monitor Setting] dialog box as follows:

- Enter ***monitor1*** in the [Name] edit box.
- Set the parameters in the [Options] group box as follows:
  (a) Use the [Watch] window to refer to the address on the line where variable `a`, which is defined within the `tutorial` function, is allocated and enter this address in the [Address] edit box. In this example, ***H'00FFB400*** is entered.
  (b) Enter ***H'20*** in the [Size (byte)] combo box.
  (c) Select BYTE (ASCII) from the [Access (Format)] combo box.
  (d) Check the [Auto-Refresh at rate (ms)] check box and enter ***D'00500*** in the edit box.
  (e) Check the [Reading the Initial Value] check box.
- Set the parameters in the [Color] group box as follows:
  (a) Select Change from the [Change Indicator] combo box.
  (b) Select red and white in the [Foreground] and [Background] combo boxes, respectively.
  (c) Check the [Mayfly] check box.

Note: Depending on the operating system in use, the foreground and background colors may not be selectable.

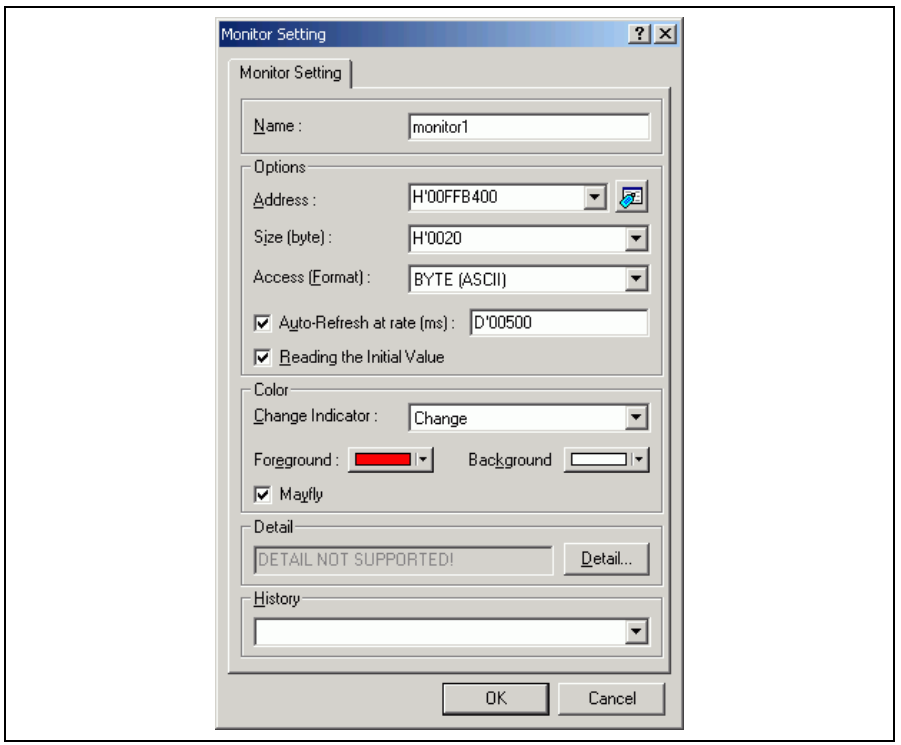


Figure 4.69 [Monitor Setting] Dialog Box (Setting Completed)

(3) Click the [OK] button to open the [Monitor] window.

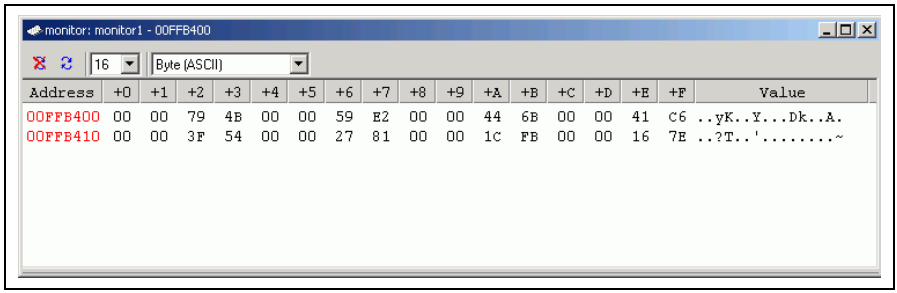


Figure 4.70 [Monitor] Window

(4) Select [Reset Go] from the [Debug] menu. When the contents of the address range changes by execution, the updated values are in red (i.e. the color that was selected in the [Foreground] and [Background] combo boxes). Values will be displayed in black if they have not been updated or a certain period of time has elapsed since the last update.

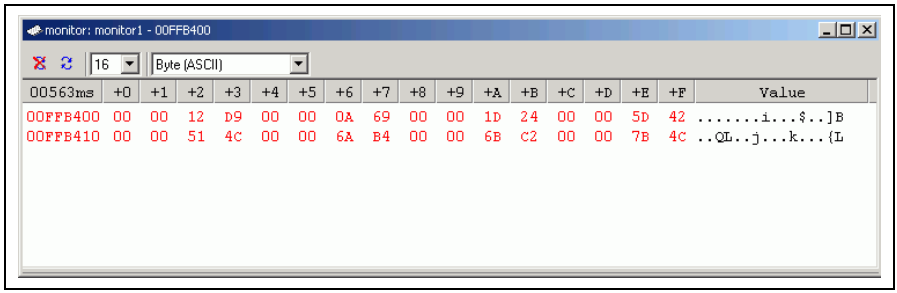


Figure 4.71 [Monitor] Window (during Execution)

(5) After you have finished checking the states in the [Monitor] window, select [Halt Program] from the [Debug] menu to halt the program's execution.

4.22 What Next?

In this tutorial, some of the main features of the emulator and the High-performance Embedded Workshop operation have been given. By using the emulation functions provided by the emulator, a high-level debugging is possible. The conditions caused by hardware and software can be accurately classified and the users can investigate the problems effectively.

Section 5 Software Specifications and Notes Specific to This Product

This section describes the software specifications and notes specific to the H8SX E6000H emulator.

5.1 Supported Hardware

This emulator software is specialized for the H8SX E6000H emulator.

5.2 Target Platforms

The following debugging platforms can be selected in this emulator. The target MCUs to be emulated depend on the selected debugging platform.

Table 5.1 Selectable Target Platforms

Debugging Platform	Description	Hardware
H8SX/1650 E6000H Emulator CPU 1600	Select this debugging platform when the H8SX/1650 E6000H emulator is in use. The H8SX/1650, H8SX/1657F or H8SX/1651 can be emulated.	HS1650EPH60H
H8SX/1653 E6000H Emulator CPU 1600	Select this debugging platform to emulate the H8SX/1653 or H8SX/1654. Note that the user system interface board (HS1653ECN61H) is required for emulation of the H8SX/1653 and H8SX/1654.	HS1650EPH60H + HS1653ECN61H
H8SX/1663 E6000H Emulator CPU 1600	Select this debugging platform to emulate the H8SX/1663 or H8SX/1664. Note that the user system interface board (HS1664ECH61H) is required for emulation of the H8SX/1663 and H8SX/1664.	HS1650EPH60H + HS1664ECH61H
H8SX/1527 E6000H Emulator CPU 1600	Select this debugging platform when the H8SX/1527 E6000H emulator is in use. The H8SX/1525, H8SX/1527, or H8SX/1582 can be emulated.	HS1527KEPH60H
H8SX/1527R E6000H Emulator CPU 1600	Select this debugging platform when the H8SX/1527R E6000H emulator is in use. The H8SX/1525R, H8SX/1527R, or H8SX/1582 can be emulated.	HS1527REPH60H
H8SX/1544 E6000H Emulator CPU 1600	Select this debugging platform when the H8SX/1544 E6000H emulator is in use. The H8SX/1544, or H8SX/1543 can be emulated.	HS1544EPH60H

5.3 Memory Map

- Some emulators may not support the memory mapping function.
- Attributes of a memory map can be defined in units of blocks as shown in figure 5.1.
- No_access Area may not be selectable as an attribute of a memory map depending on the emulator in use.
- No memory exists when No_access Area is selected, even though it is defined as an internal memory area.
- No_access Area is not selectable when the attribute of the address range selected in the [Memory Mapping] dialog box is On Chip or User.
- The minimum unit for memory mapping is one block (1 Mbyte).

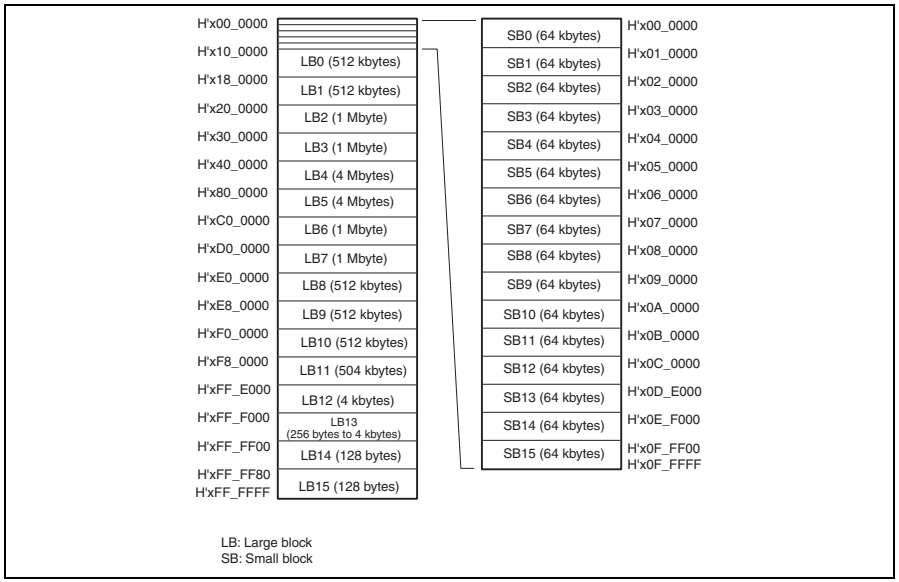


Figure 5.1 Blocks Divided in 16-Mbyte Mode

5.4 Displaying and Modifying the Contents of Memory

5.4.1 Displaying and Modifying the Contents of Memory during Execution

The emulator accesses memory in the following two ways to display and modify the contents of memory during user program execution.

Table 5.2 Access Types for Displaying and Modifying Contents of Memory

Access Type	Description	Period Suspended	Display	Modification
Monitor function	Automatically updates the display of the memory contents without stopping the user program execution. If the specified range is not accessed, the contents of memory will not be updated.	None	Enabled	Disabled
Parallel access function	Temporarily halts the user program execution. When updating, a forced memory access occurs and realtime emulation will not be performed.	51 bus cycles are required for a 4-byte access.	Enabled	Enabled

These access types have the following characteristics.

Table 5.3 Characteristics of Displaying and Modifying Contents of Memory

Access Type	Target Window	Target Memory Area
Monitor function	Realtime update of display in the [Monitor] window and in the [Watch] window when the monitor function is used.	Specified eight points or up to 256 bytes of the areas that the user program is allowed to access
Parallel access function	Windows that display the memory contents other than the [Monitor] window. Tooltip watch and instant watch. Command to display or modify the contents of memory.	All areas that the user program is allowed to access

5.4.2 Monitor Function

- Up to eight points (up to 32 bytes per point) can be specified for the monitor function.
- The monitor function is implemented by eight 32-byte hardware channels. The address range specified for one channel must be aligned to a 32-byte boundary; two channels should be used to specify a range across a 32-byte boundary. Accordingly, when multiple ranges are specified across 32-byte boundaries, the total specifiable size is less than 256 bytes.
- When monitor function conditions are set or modified during user program execution, the program is not executed in realtime.
- When [Format] is modified during user program execution, the program is not executed in realtime.

5.4.3 Parallel Access Function

The parallel access function is implemented by the DTC channels specialized for the debugger, which are not user resources. However, because the DTC channels are included in the user-resource DTC module, the parallel access function may not be available in the following cases depending on the state of the target MCU or the emulator settings:

- When the DTC module is in the module stop state, the parallel access function is not available.
- When the priority of the DTC is lower than that of the CPU, the parallel access function is not available.
- In the software standby mode, the parallel access function is not available.
- When there is a conflict between a parallel access and a reset, the parallel access fails.
- When there is a conflict between a parallel access and an NMI interrupt, the parallel access fails. Bit 0 and the transfer stop flag (ERR) of the DTC control register (DTCCR) are set.
- When an address error occurs during a parallel access, the parallel access fails. Bit 0 and the transfer stop flag (ERR) of the DTC control register (DTCCR) are set.
- When [Enable DTC parallel access] is disabled in the [Configuration Properties] dialog box, the parallel access function is not available.
- When the profiling function is used, the parallel access function during user program execution is not available.

If a parallel access to read memory fails or a parallel access is attempted under the condition where the parallel access function is not available, incorrect memory contents will be displayed. If a parallel access to write to memory fails, incorrect values will be written to.

If the message [Parallel Access Error] appears on the [Debug] sheet of the [Output] window when a parallel access fails, parallel accesses will be disabled until a break occurs.

If the parallel access function is not necessary or is not available, clear [Enable DTC parallel access] in the [Configuration Properties] dialog box to avoid unnecessary accesses.

5.5 Executing Your Program

5.5.1 Step Execution

Break conditions are ignored during step execution, but triggers will be output.

5.6 Event Functions

5.6.1 Software Breakpoints

- A software breakpoint is accomplished by replacing the instruction at the specified address with a special instruction. Accordingly, it can only be set to the RAM area including the emulation memory. However, it cannot be set to the following addresses:
  - An address whose memory content is H'5770
  - Areas other than the CS areas (except for the on-chip ROM and RAM areas)
- Do not modify the contents of the software breakpoints addresses by the user program.
- The content of a software breakpoint address is replaced by a break instruction during user program execution.
- The maximum number of software breakpoints and temporary PC breakpoints in [Temporary PC Breakpoints] of the [Run Program] dialog box is 255 in total. Therefore, when 255 software breakpoints have been set, no temporary breakpoint set in [Temporary PC Breakpoints] of the [Run Program] dialog box is valid. Ensure that the total number of software breakpoints and temporary PC breakpoints is 255 or less.
- The [Go To Cursor] function, which can be selected from the [Debug] menu or the popup menu after the mouse cursor is positioned on the target address in the [Editor] window, is implemented by a software breakpoint. Accordingly, when 255 software breakpoints have been set, this function is not available.
- Do not set a breakpoint immediately after a delayed branch instruction (at a slot instruction). If this is attempted, a slot illegal instruction interrupt will occur when the delayed branch instruction is executed, and the break will not occur.

5.6.2 On-Chip Break

- The satisfaction count can only be set for channel 4.
- A data bus condition can only be set for channel 4.
- The address and data conditions are satisfied on the bus cycles where the values on the address bus or data bus match. Consider the following points when setting these conditions.
  - Longword access

    Longword data is read and written in a single bus cycle. A data condition is only valid for a longword access when specified as longword. When a specified address is not a multiple of four (4n), several cycles will be generated as shown in tables 5.4 and 5.5. Set address and data conditions according to these tables. Note that longword data is only valid as the size of an access.
  - Word access

    Word data is read and written in a single bus cycle. Word data is only valid as the size of an access. When a specified address is not a multiple of two (2n), several cycles will be generated as shown in tables 5.4 and 5.5. Set address and data conditions according to these tables. Note that word data is only valid as the size of an access.
  - Byte access

    Byte data is read and written in a single bus cycle. A data condition is only valid for a byte access when specified as byte. Any address condition, whether an even or odd address, is valid.

5.6.3 On-Emulator Break

- A break will occur several cycles after a condition is satisfied.
- The states of IRQ15 to IRQ0 are ORed and this result is applied as the IRQ condition.
- The address and data conditions are satisfied on the bus cycles where the values on the address bus or data bus match. Consider the following points when setting these conditions.
    - Longword access

      Longword data is read and written in a single bus cycle. A data condition is only valid for a longword access when specified as longword. When a specified address is not a multiple of four (4n), several cycles will be generated as shown in tables 5.4 and 5.5. Set address and data conditions according to these tables. Note that longword data is only valid as the size of an access.
    - Word access

      Word data is read and written in a single bus cycle. Word data is only valid as the size of an access. When a specified address is not a multiple of two (2n), several cycles will be generated as shown in tables 5.4 and 5.5. Set address and data conditions according to these tables. Note that word data is only valid as the size of an access.
    - Byte access

      Byte data is read and written in a single bus cycle. A data condition is only valid for a byte access when specified as byte. Any address condition, whether an even or odd address, is valid.

Use the mask function so that no invalid data of a 32-bit data bus will be applied as a condition to search data.

Table 5.4 User Bus and Break Setting for an Output of Data (Big Endian)

Access size	Address	Break	area bus		
		32 bit	8 bit	16 bit	32 bit
Byte	00				
	01				
	10				
	11				
Word	00				
	01				
	10				
	11				
Long word	00				
	01				
	10				
	11				

Table 5.5 User Bus and Break Setting for an Output of Data (Little Endian)

Access size	Address	Break	area bus		
		32 bit	8 bit	16 bit	32 bit
Byte	00				
	01				
	10				
	11				
Word	00				
	01				
	10				
	11				
Long word	00				
	01				
	10				
	11				

5.7 Trace Functions

5.7.1 Displaying the Trace Information

- The states of IRQ15 to IRQ0 are ORed and this result is applied as the IRQ condition.
- The executed instructions and timestamp cannot be displayed together.

5.7.2 Specifying Trace Acquisition Conditions

- The trace will stop several cycles after a condition is satisfied.
- The state of IRQ15 to IRQ0 are ORed and this result is applied as the IRQ condition.
- Six or more bus cycles are required between pass points of sequential trace stop conditions and reset condition.
- Six or more bus cycles are required between the start and end of measurement when [1-2], [3-4], or [5-6] is specified in [Run time count between trigger outputs] of the [Trace Acquisition Properties] dialog box.
- Fifteen or more bus cycles are required between the start and end of measurement when [7-8] is specified in [Run time count between trigger outputs] of the [Trace Acquisition Properties] dialog box.
- Six or more bus cycles are required between trace stop mode conditions.
- A sequential break or a trace stop may be incorrect when the user program is executed after the specified address condition has been applied as the PC address to start execution.
- The Point to Point trace mode is not available when channel 1 is used for the performance analysis function.

5.7.3 Searching for a Trace Record

- When the range for searching is specified in the [General] page, a PTR value to end the search can be specified in the [Start PTR] option, and a PTR value to start the search can be specified in the [End PTR].
- When the user clock (i.e. system clock signal ϕ) has been selected in [Time measurement unit] of the [Trace Acquisition Properties] dialog box, no time stamp information will be searched.

5.7.4 Filtering Trace Records

- After the trace information is filtered, all trace information displayed in the [Trace] window is saved; a range for saving trace information cannot be specified. To save a specific range of trace information, the filter range must be specified in the [General] page of the [Trace Filter] dialog box.
- When the user clock (i.e. system clock signal ϕ) has been selected in [Time measurement unit] of the [Trace Acquisition Properties] dialog box, no time stamp information will be filtered.

5.8 Monitor Function

The foreground and background colors cannot be changed in some operating systems.

5.9 Performance Analysis Function

5.9.1 Errors

An error will be included in the measured performance as follows:

- ±one-resolution error (±20-ns error when the measurement resolution is 20 ns)
  This error may occur when the user program execution starts or stops (breaks) or when the measurement start or end condition is satisfied.
- Frequency stability of the crystal oscillating module for performance analysis: ±0.01%

5.9.2 Notes

- In all measurement modes, the interval between the end condition satisfaction and the next start condition satisfaction must be longer than one-measurement-resolution time. If the interval is shorter than that, the interval itself is included in the measured time.
- In [Time Of Specified Range Measurement], measurement stops when an instruction is fetched outside the specified range. In [Start Point To End Point Measurement] and [Start Range To End Range Measurement], measurement stops when the specified end condition is satisfied. When the same addresses are specified for these modes, the time measured in [Time Of Specified Range Measurement] is longer than that measured in [Start Point To End Point Measurement] or [Start Range To End Range Measurement].
- Execution time is measured by using address bus values in prefetch cycles. If the end address condition is specified at an address near the instruction following a branch instruction or delayed slot instruction, correct time cannot be measured. Check the bus trace display for the operation after the branch instruction prefetch cycle, and specify the end address condition at the address in a prefetch cycle which will not be executed by the branching.
- Channel 1 is not available for performance analysis when the Point to Point trace mode is selected.
- The resolution for the performance analysis function can be set in [Timer Resolution] of the [Configuration Properties] dialog box. If the clock counter value is set as the resolution, the value shown in [RUN-TIME] and [MAX-MIN-TIME] will be that of the clock counter (displayed in hexadecimal).

5.10 Profiling Function

- If there is no stack information file (extension is '.sni') that is output from the optimizing linkage editor, only the functions that have been executed during the profiling data measurement are displayed. For details of the stack information file, refer to the manual of the optimizing linkage editor.
- The stack size differs from the actual value. It should be used as a reference value during a function call. If there is no stack information file (extension is '.sni') that is output from the optimizing linkage editor, the stack size is not displayed.
- While the profiling function is used, software break and on-emulator break, which are event functions, are not available.
- While the profiling function is used, the parallel access function during user program execution is not available.
- Since the profiling function internally breaks user program execution, the program is not executed in realtime. The measured value includes an error.

5.11 RTOS Extension Function

5.11.1 Input Values for the [TASK ID writing area] Input Edit Box

Specify the address to which the task IDs are written. Set an integer multiple of eight within the on-chip I/O area that does not contain I/O registers. The address ranges for the on-chip I/O area are H'00FFEA00 to H'00FFFE00 and H'00FFFF20 to H'00FFFFFC.

Table 5.6 shows examples of input values for the [TASK ID writing area] input edit box for particular debugging platforms.

Table 5.6 Examples of Input Values for the [TASK ID writing area] Input Edit Box

Debugging Platform	Example of Input Value for the [TASK ID writing area] Input Edit Box
H8SX/1650 E6000H Emulator CPU 1600	H'00FFFBE8
H8SX/1653 E6000H Emulator CPU 1600	H'00FFFBE8
H8SX/1663 E6000H Emulator CPU 1600	H'00FFFBE8
H8SX/1527 E6000H Emulator CPU 1600	H'00FFFBE8
H8SX/1527R E6000H Emulator CPU 1600	H'00FFFBE8
H8SX/1544 E6000H Emulator CPU 1600	H'00FFFBE8

5.11.2 Debugging Area for the E6000H Emulator

For the debugging area for the E6000H emulator, refer to section 5, Configuration, in the HI1000/4 User's Manual.

5.11.3 Note on Using the HI1000/4 Debugging Extension

When using the HI1000/4 debugging extension, select [Don't Care] under [RTOS Support Function] in the [RTOS Support Function Configuration Properties] dialog box to disable the RTOS extension function.

5.12 Input Format

5.12.1 Entering Masks

Address bus conditions and data bus conditions can be input with masks. Addresses can be masked in 1-, 3-, or 4-bit units. When a bit is masked, it always satisfies the condition.

To specify a mask for an address bus condition, specify the mask value in the [Mask] area.

The mask for data conditions is similarly specified in the [Mask] area.

To specify any further mask, specify 1 for the digits to be ignored. Examples of mask specification are listed below.

Table 5.7 Address Mask Specification

No	Input Value	Mask Unit	Example	Masked Bits
1	Binary	1 bit	B'00000111	Masks bits 0 to 2
2	Octal	3 bits	O'000017	Masks bits 0 to 3 and 5
3	Hexadecimal	4 bits	H'07FF	Masks bits 0 to 10

Section 6 Error Messages

6.1 Error Messages of the Emulator

6.1.1 Error Messages at Emulator Initiation

The emulator displays error messages in the format below if an error occurs at emulator initiation in the dedicated message dialog box when the High-performance Embedded Workshop is used.

Table 6.1 lists error messages at emulator initiation.

Table 6.1 Error Messages at Initiation

Error Message	Description and Solution
There is no configuration file.	The configuration file that is required to initiate the emulator cannot be found. Exit and re-install the High-performance Embedded Workshop. Then re-connect the user system interface cable, turn on the power of the emulator, and re-initiate the High-performance Embedded Workshop. If the problem is not solved, contact us and describe the error occurrence in detail.
The contents of the configuration file are incorrect.	The configuration file that is required to initiate the emulator is invalid. Exit and re-install the High-performance Embedded Workshop. Then re-connect the user system interface cable, turn on the power of the emulator, and re-initiate the High-performance Embedded Workshop. If the problem is not solved, contact us and describe the error occurrence in detail.
Main Board not Support (XX XX XX) Emulator is switched off or not connected	The emulator power is not turned on, or the user system interface cable is not connected. Exit the High-performance Embedded Workshop, re-connect the user system interface cable, turn on the power of the emulator, and re-initiate the High-performance Embedded Workshop. If the problem is not solved, contact us and describe the error occurrence in detail.
Emulation Board not Support (XX XX XX) Emulator is switched off or not connected	The emulator power is not turned on, or the user system interface cable is not connected. Exit the High-performance Embedded Workshop, re-connect the user system interface cable, turn on the power of the emulator, and re-initiate the High-performance Embedded Workshop. If the problem is not solved, contact us and describe the error occurrence in detail.
EVA chip Board not Support (XX XX XX) Emulator is switched off or not connected	The emulator power is not turned on, or the user system interface cable is not connected. Exit the High-performance Embedded Workshop, re-connect the user system interface cable, turn on the power of the emulator, and re-initiate the High-performance Embedded Workshop. If the problem is not solved, contact us and describe the error occurrence in detail.
Can't initialize G/A registers	An error occurred during the initialization of the emulator. Exit the High-performance Embedded Workshop, re-connect the user system interface cable, turn on the power of the emulator, and re-initiate the High-performance Embedded Workshop. If the problem is not solved, contact us and describe the error occurrence in detail.
There is no effective clock source	A valid clock source cannot be found. Connect a valid clock source.
This mode can not specify	The state of mode pins for the target board is incorrect. Initiation is only possible in mode 4. Set the mode pins correctly.

Table 6.1 Error Messages at Initiation (cont)

Error Message	Description and Solution
Can't find firmware file Firmware open Error Firmware Download Error Firmware Name Error	There is an error in the file that is required at emulator initiation. Exit the High-performance Embedded Workshop, re-connect the user system interface cable, turn on the power of the emulator, and re-initiate the High-performance Embedded Workshop. If the problem is not solved, contact us and describe the error occurrence in detail.
Failed to receive a firmware initialization command.	Initiation of the emulator firmware has failed. Exit the High-performance Embedded Workshop, re-connect the user system interface cable, turn on the power of the emulator, and re-initiate the High-performance Embedded Workshop. If the problem is not solved, contact us and describe the error occurrence in detail.
Target system is Vcc down	The value of Vcc is lower than the specified threshold value.

6.1.2 Error Messages during Emulation

The emulator displays error messages if an error occurs during emulation in the dedicated message dialog box when the High-performance Embedded Workshop is used, and on the status bar. Table 6.2 lists error messages during emulation.

Table 6.2 Error Messages during Emulation

Error Message	Description and Solution
Communication DLL error. Communication Timeout error.	The power of the emulator is turned off or there is a communication error. Exit the High-performance Embedded Workshop, re-connect the user system interface cable, turn on the power of the emulator, and re-initiate the High-performance Embedded Workshop. If the problem is not solved, contact us and describe the error occurrence in detail.
Parallel Access Error	An error has occurred during a parallel access. Parallel accesses are disabled until a break occurs.

Appendix A Menus

Table A.1 shows GUI menus.

Table A.1 GUI Menus

Menu	Option			Shortcut	Toolbar Button	Remarks
View	Command Line			Ctrl + L		Opens the [Command Line] window.
	Workspace			Alt + K		Opens the [Workspace] window.
	Output			Alt + U		Opens the [Output] window.
	Disassembly			Ctrl + D		Opens the [Disassembly] window.
	CPU	Registers		Ctrl + R		Opens the [Register] window.
		Memory…		Ctrl + M		Opens the [Memory] window.
		IO		Ctrl + I		Opens the [IO] window.
		Status		Ctrl + U		Opens the [Status] window.
		Extended Monitor				Opens the [Extended Monitor] window.
		Monitor	Monitor Setting…	Shift + Ctrl + E		Opens the [Monitor] window.
			Windows Select…			Opens the [Windows Select] dialog box to list, add, or edit the [Monitor] window.
	Sym-bol	Labels		Shift + Ctrl + A		Opens the [Labels] window.
		Watch		Ctrl + W		Opens the [Watch] window.
		Locals		Shift + Ctrl + W		Opens the [Locals] window.
	Code	Eventpoints		Ctrl + E		Opens the [Event] window.
		Trace		Ctrl + T		Opens the [Trace] window.
		Code Coverage...		Shift + Ctrl + H		Opens the [Code Coverage] window.
		Data coverage...		Shift + Ctrl + Z		Opens the [Data Coverage] window.
		Stack Trace		Ctrl + K		Opens the [Stack Trace] window.

Table A.1 GUI Menus (cont)

Menu	Option		Shortcut	Toolbar Button	Remarks
View (cont)	Graphic	Image…	Shift + Ctrl + G		Opens the [Image] window.
		Waveform…	Shift + Ctrl + V		Opens the [Waveform] window.
	Performance	Performance Analysis	Shift + Ctrl + P		Opens the [Performance Analysis] window.
		Profile	Shift + Ctrl + F		Opens the [Profile] window.

Menu	Option	Shortcut	Toolbar Button	Remarks
Debug	Debug Sessions…			Opens the [Debug Sessions] dialog box to list, add, or remove the debug session.
	Debug Settings…			Opens the [Debug Settings] dialog box to set the debugging conditions or download modules.
	Reset CPU			Resets the target hardware and sets the PC to the reset vector address.
	Go	F5		Starts executing the user program at the current PC.
	Reset Go	Shift + F5		Resets the target hardware and executes the user program from the reset vector address.
	Go To Cursor			Starts executing the user program at the current PC until the PC reaches the address indicated by the current text cursor position.
	Set PC To Cursor			Sets the PC to the address at the row of the text cursor.
	Run…			Launches the [Run Program] dialog box allowing the user to enter the PC or PC breakpoint during executing the user program.
	Display PC	Shift+Ctrl +Y		Opens the [Editor] or [Disassembly] window at the address of the PC.

Table A.1 GUI Menus (cont)

Menu	Option		Shortcut	Toolbar Button	Remarks
Debug (cont)	Step In		F11		Executes a block of user program before breaking.
	Step Over		F10		Executes a block of user program before breaking. If a subroutine call is reached, then the subroutine will not be entered.
	Step Out		Shift + F11		Executes the user program to reach the end of the current function.
	Step…				Launches the [Step Program] dialog box allowing the user to modify the settings for stepping.
	Step Mode	Auto			Steps only one source line when the [Editor] window is active. When the [Disassembly] window is active, stepping is executed in a unit of assembly instructions.
		Assembly			Executes stepping in a unit of assembly instructions.
		Source			Steps only one source line.
	Halt Program		Esc		Stops the execution of the user program.
	Initialize				Disconnects the emulator and connects it again.
	Connect				Connects the emulator.
	Disconnect				Disconnects the emulator.
	Save Memory…				Saves the specified memory area data to a file.
	Verify Memory…				Verifies file contents against memory contents.
	Configure Overlay…				Selects the target section group when the overlay function is used.
	Download Modules				Downloads the object program.
	Unload Modules				Unloads the object program.

Table A.1 GUI Menus (cont)

Menu	Option		Shortcut	Toolbar Button	Remarks
Setup	Customize…				Customize the High-performance Embedded Workshop application.
	Options…				Sets option of the High-performance Embedded Workshop application.
	Format Views…				Configure fonts, colors, keywords and so on, for the window.
	Radix	Hexadecimal			Uses a hexadecimal for displaying a radix in which the numerical values will be displayed and entered by default.
		Decimal			Uses a decimal for displaying a radix in which the numerical values will be displayed and entered by default.
		Octal			Uses an octal for displaying a radix in which the numerical values will be displayed and entered by default.
		Binary			Uses a binary for displaying a radix in which the numerical values will be displayed and entered by default.
	Emu-lator	System…			Opens the [Configuration Properties] dialog box allowing the user to modify the emulator settings.
		Memory Resource…			Opens the [Memory Mapping] dialog box allowing the user to view and edit the emulator 's current memory map.
		RTOS Support Function			Opens the [RTOS Support Function Configuration Properties] dialog box allowing the user to make settings for debugging of RTOS tasks.

Appendix B Command Lines

Table B.1 lists the High-performance Embedded Workshop commands.

Table B.1 High-performance Embedded Workshop Commands

No.	Command Name	Abbreviation	Function
1	!	-	Comment
2	ADD_FILE	AF	Adds a file to the current project
3	ANALYSIS	AN	Enables or disables performance analysis
4	ANALYSIS_RANGE	AR	Sets or displays a performance analysis range
5	ANALYSIS_RANGE_ DELETE	AD	Deletes a performance analysis range
6	ASSEMBLE	AS	Assembles instructions into memory
7	ASSERT	-	Checks if an expression is true or false
8	AUTO_COMPLETE	AC	Enables or disables the auto-complete function
9	BREAKPOINT_ONCHIP	BC	Displays on-chip breakpoints, sets sequential breaks, and sets PtoP time measurement
10	BREAKPOINT_ONCHIPn	BCn	Sets on-chip breakpoint of each channel
11	BREAKPOINT_ONCHIP _CLEAR	BCC	Clears on-chip breakpoints
12	BREAKPOINT_ONCHIP _ENABLE	BCE	Enables or disables an on-chip breakpoint
13	BREAKPOINT_ ONEMULATOR	BE	Displays on-emulator breakpoints
14	BREAKPOINT_ ONEMULATORn	BEn	Sets on-emulator breakpoint of each channel
15	BREAKPOINT_ ONEMULATOR_CLEAR	BEC	Clears on-emulator breakpoints
16	BREAKPOINT_ ONEMULATOR_ ENABLE	BEE	Enables or disables an on-emulator breakpoint
17	BREAKPOINT_ SOFTWARE	BS	Sets a software breakpoint
18	BREAKPOINT_ SOFTWARE_CLEAR	BSC	Clears software breakpoints
19	BREAKPOINT_ SOFTWARE_ENABLE	BSE	Enables or disables a software breakpoint
20	BUILD	BU	Performs a build on the current project
21	BUILD_ALL	BL	Performs a build all on the current project
22	CHANGE_ CONFIGURATION	CC	Sets the current configuration
23	CHANGE_PROJECT	CP	Sets the current project
24	CHANGE_SESSION	CS	Changes the current session
25	CLOSE_WORKSPACE	CW	Close the current workspace
26	CONFIGURE_ PLATFORM	CPF	Sets the debugging environment for the emulator
27	CUSTOM_DEVICE	CDE	Customizes the device
28	DEFAULT_OBJECT_ FORMAT	DO	Sets the default object (program) format
29	DEVICE_TYPE	DE	Selects a device type to emulate

Table B.1 High-performance Embedded Workshop Commands (cont)

No.	Command Name	Abbreviation	Function
30	DISASSEMBLE	DA	Disassembles memory contents
31	EMULATOR_CLOCK	ECK	Selects the clock rate of the target MCU for the emulator
32	ERASE	ER	Clears the [Command Line] window
33	EVALUATE	EV	Evaluates an expression
34	EXMONITOR_DISPLAY	EXMD	Displays the content of the expansion monitor
35	EXMONITOR_SET	EXMS	Selects whether or not to display the items in the expansion monitor
36	EXMONITOR_ SETRATE	EXMSR	Sets the time to update the expansion monitor during emulation or a break
37	FILE_LOAD	FL	Loads an object (program) file
38	FILE_SAVE	FS	Saves memory to a file
39	FILE_UNLOAD	FU	Unloads a file
40	FILE_VERIFY	FV	Verifies file contents against memory
41	GENERATE_MAKE_ FILE	GM	Creates a makefile to be built outside the High-performance Embedded Workshop
42	GO	GO	Executes user program
43	GO_RESET	GR	Executes user program from reset vector
44	GO_TILL	GT	Executes user program until temporary breakpoint
45	HALT	HA	Halts the user program
46	HELP	HE	Displays the command line help
47	INITIALIZE	IN	Initializes the debugging platform
48	LOG	LO	Controls command output logging
49	MAP_DISPLAY	MA	Displays memory mapping
50	MAP_SET	MS	Sets memory mapping
51	MEMORY_COMPARE	MC	Compares memory contents
52	MEMORY_DISPLAY	MD	Displays memory contents
53	MEMORY_EDIT	ME	Modifies memory contents
54	MEMORY_FILL	MF	Modifies the content of a memory area by specifying data
55	MEMORY_FIND	MI	Searches for data within the memory range
56	MEMORY_MOVE	MV	Moves a block of memory
57	MEMORY_TEST	MT	Tests a block of memory
58	MODE	MO	Sets or displays the MCU mode
59	MONITOR_CLEAR	MOC	Deletes a monitor point
60	MONITOR_DISPLAY	MOD	Displays the content of the monitor
61	MONITOR_REFRESH	MOR	Controls an automatic update of the content of the monitor
62	MONITOR_SET	MOS	Sets or displays a monitor point
63	OPEN_WORKSPACE	OW	Opens a workspace
64	PROFILE	PR	Enables or disables the profile
65	PROFILE_DISPLAY	PD	Displays profiling results
66	PROFILE_SAVE	PS	Saves profiling results
67	QUIT	QU	Exits High-performance Embedded Workshop
68	RADIX	RA	Sets default input radix

Table B.1 High-performance Embedded Workshop Commands (cont)

No.	Command Name	Abbreviation	Function
69	REFRESH	RF	Updates windows related to memory
70	REGISTER_DISPLAY	RD	Displays CPU register values
71	REGISTER_SET	RS	Sets CPU register contents
72	REMOVE_FILE	REM	Deletes the specified file from the current project
73	RESET	RE	Resets CPU
74	RTOS	RT	Sets debugging of RTOS tasks
75	SAVE_SESSION	SE	Saves the current session
76	SLEEP	-	Delays command execution
77	STEP	ST	Steps program (by instructions or source lines)
78	STEP_MODE	SM	Sets the step mode
79	STEP_OUT	SP	Steps out of the current function
80	STEP_OVER	SO	Steps program, not stepping into functions
81	STEP_RATE	SR	Sets or displays rate of stepping
82	SUBMIT	SU	Executes a command file
83	SYMBOL_ADD	SA	Defines a symbol
84	SYMBOL_CLEAR	SC	Deletes a symbol
85	SYMBOL_LOAD	SL	Loads a symbol information file
86	SYMBOL_SAVE	SS	Saves a symbol information file
87	SYMBOL_VIEW	SV	Displays symbols
88	STATUS	STS	The content of the [Platform] sheet in the [Status] window is displayed.
89	SAVE_WORKSPACE	SW	Saves the current workspace
90	TCL	-	Enables or disables the TCL
91	TIMER	TI	Sets or displays the timer resolution
92	TOOL_INFORMATION	TO	Outputs information on the currently registered tool to a file
93	TRACE	TR	Displays trace information
94	TRACE_ACQUISITION	TA	Sets or displays trace acquisition parameters
95	TRACE_ACQUISITIONn	TAn	Sets PtoP point and each channel for trace acquisition conditions
96	TRACE_ACQUISITION_CLEAR	TAC	Deletes trace acquisition parameters
97	TRACE_BINARY_COMPARE	TBC	Compares a trace binary file with the current trace information
98	TRACE_BINARY_SAVE	TBV	Outputs trace information into a binary file
99	TRACE_FILTER	TF	Filters trace information
100	TRACE_SAVE	TV	Outputs trace information into a file
101	TRACE_STATISTIC	TST	Analyzes statistic information
102	UPDATE_ALL_DEPENDENCIES	UD	Updates dependencies of the current project
103	USER_SIGNALS	US	Enables or disables the user signal information

Table B.1 High-performance Embedded Workshop Commands (cont)

No.	Command Name	Abbreviation	Function
104	WATCH_ADD	WA	Adds a watch item
105	WATCH_AUTO_UPDATE	WU	Selects or cancels automatic update of watch items
106	WATCH_DELETE	WD	Deletes a watch item
107	WATCH_DISPLAY	WI	Displays the contents of the Watch window
108	WATCH_EDIT	WE	Edits the value of a watch item
109	WATCH_EXPAND	WX	Expands or collapses a watch item
110	WATCH_RADIX	WR	Changes the radix of a watch item to be displayed
111	WATCH_SAVE	WS	Saves the contents of the Watch window to a file

For the syntax of each command, refer to the online help.

Renesas Microcomputer Development Environment System
User's Manual
H8SX E6000H Emulator

Publication Date:	Rev.1.00, August 25, 2003
	Rev.9.00, June 13, 2006
Published by:	Sales Strategic Planning Div.
	Renesas Technology Corp.
Edited by:	Customer Support Department
	Global Strategic Communication Div.
	Renesas Solutions Corp.

© 2006. Renesas Technology Corp., All rights reserved. Printed in Japan.

Renesas Technology Corp. Sales Strategic Planning Div. Nippon Bldg., 2-6-2, Ohte-machi, Chiyoda-ku, Tokyo 100-0004, Japan

RENESAS SALES OFFICES

http://www.renesas.com

Refer to "**http://www.renesas.com/en/network**" for the latest and detailed information.

Renesas Technology America, Inc.
450 Holger Way, San Jose, CA 95134-1368, U.S.A
Tel: <1> (408) 382-7500, Fax: <1> (408) 382-7501

Renesas Technology Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K.
Tel: <44> (1628) 585-100, Fax: <44> (1628) 585-900

Renesas Technology (Shanghai) Co., Ltd.
Unit 204, 205, AZIACenter, No.1233 Lujiazui Ring Rd, Pudong District, Shanghai, China 200120
Tel: <86> (21) 5877-1818, Fax: <86> (21) 6887-7898

Renesas Technology Hong Kong Ltd.
7th Floor, North Tower, World Finance Centre, Harbour City, 1 Canton Road, Tsimshatsui, Kowloon, Hong Kong
Tel: <852> 2265-6688, Fax: <852> 2730-6071

Renesas Technology Taiwan Co., Ltd.
10th Floor, No.99, Fushing North Road, Taipei, Taiwan
Tel: <886> (2) 2715-2888, Fax: <886> (2) 2713-2999

Renesas Technology Singapore Pte. Ltd.
1 Harbour Front Avenue, #06-10, Keppel Bay Tower, Singapore 098632
Tel: <65> 6213-0200, Fax: <65> 6278-8001

Renesas Technology Korea Co., Ltd.
Kukje Center Bldg. 18th Fl., 191, 2-ka, Hangang-ro, Yongsan-ku, Seoul 140-702, Korea
Tel: <82> (2) 796-3115, Fax: <82> (2) 796-2145

Renesas Technology Malaysia Sdn. Bhd
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No.18, Jalan Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: <603> 7955-9390, Fax: <603> 7955-9510

H8SX E6000H Emulator User's Manual

Renesas Technology Corp.
2-6-2, Ote-machi, Chiyoda-ku, Tokyo, 100-0004, Japan